Final Report
Contract NAS 8-28425

Targetting and Guidance Program Documentation

by

E. F. Harrold and J. F. Neyhard

Submitted July 15, 1974 to
National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center
Huntsville, Alabama 35812

by

IBM Federal Systems Division
18100 Frederick Pike
Gaithersburg, Maryland 20760

Work performed at

Advanced Systems Design Department
3 New England Executive Park
Burlington, Mass. 01803

(NASA-CR-120393) TARGETTING AND GUIDANCE PROGRAM DOCUMENTATION Final Report (International Business Machines Corp.) 61 p HC $6.25 CSCL 22A
N74-31320
Unclas
G3/30 46510

TABLE OF CONTENTS

Introduction

A FORTRAN computer program has been developed which automatically targets two and three burn rendezvous missions and performs feedback guidance using the previously developed GUIDE algorithm. The program was designed to accept a large class of orbit specifications and automatically chooses a two or three burn mission depending upon the time alignment of the vehicle and target. The orbits may be specified as any combination of circular and elliptical orbits and may be coplanar or inclined, but must be aligned coaxially (i.e. line of intersection of orbital planes and orbital major axes coincident) with their perigees in the same direction. The program accomplishes the required targetting by repeatedly converging successively more complex missions. It solves the coplanar impulsive version of the mission, then the finite burn coplanar mission and finally the full plane change mission. The GUIDE algorithm is exercised in a feedback guidance mode by taking the targeted solution and moving the vehicle state step by step ahead in time adding acceleration and navigational errors and reconverging from the perturbed states at fixed guidance update intervals.

The targetting and guidance algorithm converges all two burn missions easily and exhibits good guidance behavior for these missions. Three burn missions were much more sensitive and required special loops to insure convergence. The outbound three burn mission had to be converged backwards in time and plane change was most readily incorporated by eliminating the third burn and solving the appropriate two burn mission, reintroducing the third burn at the end. In a targetting mode these techniques cause no particular problem and insure convergence. In guidance mode the convergence problems are more difficult to compensate for and may limit real time use. The program as it now stands attempts to optimize over all three burns and although it has maintained convergence for all missions attempted, the guidance corrections have been larger than desired. In the future it may be necessary to solve the guidance problem over the first burn as a rendezvous with the desired phasing or transfer orbit and to only introduce the third burn after completion of the first one.

Another study that needs to be undertaken is to optimize the soft constraint weights using the Monte Carlo capability built into the program. By altering the weights and noting the tradeoffs made between burn time and orbital injection error, a better estimate of optimal soft constraint weights can be obtained.

The remainder of this document describes the targetting and guidance program in detail, giving an overview of the program control and organization, a summary of program inputs and outputs and a detailed description of each of the subparts of the program. Also included in the document is a description of the GUIDE subroutine BVAL5, which was altered to incorporate the soft constraint formulation, and is fully documented. The other GUIDE subroutines are essentially the same as the ones described in the GUIDE 71/6 document[1] and are not described here.

[1] Cohen, A.O., "Guide 71/6 Program Documentation", IBM Federal Systems Division, Burlington, Mass., October 4, 1971.

Program Overview

The program is controlled by routine MAIN, which oversees the impulsive targetting, the convergence of the orbital transfer, and the feedback guidance. The impulsive targetting is accomplished by first determining the elements of both orbits, then defining the transfer orbit and phasing orbit (3 burn only) and determining the velocities at apogee and perigee of each orbit. Next the delta v's are calculated and the burn and coast times calculated. The transfer orbit is chosen to be tangent at both end points to the principal orbits, and the mission is classified as inbound or outbound depending on whether apogee of the final (target) orbit is less than or greater than apogee of the initial orbit. The phasing orbit is chosen to lie as close as possible to the one which results from splitting the burn at perigee into two equal halves. A closed-form solution is used for initial costate.

The converged finite-burn solution is arrived at by repeatedly converging successively more complex missions, starting with a planar mission and gradually adding in the plane change required (10° steps). To maintain convergence for outbound 3-burn missions, it was necessary to rearrange each mission and converge it in a backwards fashion, from the target orbit to the vehicle (initial) orbit. The plane change mentioned above was facilitated by changing the 3-burn mission to a 2-burn mission where the planar-converged phasing orbit was substituted for the closer orbit. After converging the 2-burn mission with the total plane change, the 3-burn mission was reinstated and converged. Finally, the 3-burn outbound mission is turned around to its normal mode and reconverged.

After targetting has been done, the guidance portion of the program is run in a feedback mode, in which it is made to respond to simulated perturvations. The routine MAIN calls BCBCB or CBCB to propagate the vehicle along each arc of the mission, and Monte Carlo statistics are collected at appropriate points and summarized at the end.

Further details of the operation of the program, as well as the routines employed, are described in the pages which follow.

User's Guide

The program is set up using NAMELIST input for ease of operation. This allows default parameter values to be specified and reduces the amount of input necessary for program execution. Typical space tug vehicle parameters are hard coded as default values and tug missions can be performed by simply specifying the desired initial and final orbits. The basic program philosophy is to use the orbital definitions to define whether the mission will be two or three burns. If the mission is circular to circular coplanar, or if the orbital elements are defined with no positions along the orbits given, or if the positions of the vehicle and target allow a two burn rendezvous, a two burn orbital transfer will be defined. Under all other conditions three burn transfers will be used. The integer NOTARG is used to control which portions of the program are executed. If NOTARG=-1 only targetting is performed. If NOTARG=1 a converged solution for the orbital transfer is read in using NAMELIST NAMSL2 and only the feedback guidance part of the program will be executed. If NOTARG is any other value both the targetting and guidance will be performed. The inputs and outputs and individual subroutines will be described in detail in the sections which follow.

Program Inputs

The program inputs are broken into three basic groups: those which define the vehicle's capabilities, those used to specify the initial and final orbits, and those used to define the Monte Carlo and perturbation parameters needed for feedback guidance evaluation.

A. Vehicle Constants

The following parameters are used to specify the vehicle, and must be in metric units. If specific impulse is inputted it is used to calculate mass rate. The default values for the parameters are typical of a space tug configuration.

Name	Symbol	Definition	Default Value
AMØ	m_0	Initial vehicle mass in kg	28803.1155 kg (63500 lbs)
THRUST	T	Thrust in kilo-Newtons	66.7233 kn (15000 lbs)

Name	Symbol	Definition	Default Value
SPFIMP	I_{sp}	Specific impulse in seconds	440 sec
AMDOT	$\dot{m}$	Mass rate in kg/sec	15.4634 kg/sec

B. Orbit Specifications

The vehicle and target orbits may be specified in four separate ways listed as sets 1-4 below. (It is assumed that both will be specified in the same fashion.) For all of the orbital definitions the perigee directions must be equal and coincident with the line of intersection of the orbital planes. If sets 2, 3 or 4 are used to specify the orbits, these conditions are satisfied automatically due to the way the orbital positions and coordinate systems are defined. If position and velocity vectors and times (set 1) are specified, the program will test to see that the conditions are satisfied and will stop if the proper perigee and line of nodes alignment is not found. When set 1 is used to specify the data the relative inclination between orbits is measured from vehicle to target orbit at perigee. In all other cases relative inclination is set by the input data. If sets 2, 3 or 4 are used to specify the orbits and the true anomalies (TANOMØ and TANOMT) are greater than or equal to zero, they will be used to specify the orbital positions. If true anomalies are not specified and TØ and TT are greater than or equal to zero they will be assumed to be mean anomalies and used to specify the orbital positions. If neither of the anomalies are specified, the orbital positions will be arbitrarily chosen to allow a two burn rendezvous. If no complete set of vehicle and target orbital data is available, the program will print the existing data and stop.

	Name	Symbol	Definition	Default Value
SET 1	RØ(3),VØ(3),TØ	$\bar{r}_0,\bar{v}_0,t_0$	Position (km) and velocity (km/sec) vectors at t_0 for vehicle orbit	TØ=-1
	RT(3), VT(3), TT	$\bar{r}_t,\bar{v}_t,t_t$	Position (km) and velocity (km/sec) vectors at t_t for target orbit	TT=-1

Set	Name	Symbol	Definition	Default Value
SET 2	AØ, EØ	a_0, e_0	Semi-major axis (km) and eccentricity for vehicle orbit	0, -1
	AT, ET	a_t, e_t	Semi-major axis (km) and eccentricity for target orbit	0, -1
	RELINC	i	Signed relative inclination (deg) as measured from vehicle to target orbit	0
	TANOMØ,TANOMT*	f	True anomalies (not required) (deg)	-1
	TØ, TT *	M	Mean anomalies if true anomalies not specified (not required) (sec)	-1

* described more fully in text above

Set	Name	Symbol	Definition	Default Value
SET 3	HAPØ, HPGØ	-	Height at apogee and perigee for vehicle orbit (km)	None
	HAPT, HPGT	-	Height at apogee and perigee for target orbit (km)	None
	RELINC	i	Same as set 2	0
	TØ, TT, TANOMØ, TANOMT		Same as set 2	-1

Set	Name	Symbol	Definition	Default Value
SET 4	RØMAG, VØMAG, FLTØ	$\lvert R_0 \rvert$, $\lvert V_0 \rvert$, α_0	Magnitude of position and velocity vectors (km) and flight angle between them for vehicle orbit	ROMAG: -1 FLTØ: -1
	RTMAG,VTMAG, FLTT	$\lvert R_t \rvert$ $\lvert V_t \rvert$ α_t	Same for target orbit	FLTT: -1
	RELINC		Same as set 2	0
	TØ, TT, TANOMØ,TANOMT		Same as set 2	-1

C. Feedback Guidance Parameters

In order to exercise the feedback guidance portion of the program and collect statistics on performance, the magnitude of the navigation update errors at the start of the first coast, at the start of the second coast and in the middle of the last burn need to be specified. The time between guidance updates on coast and burn arcs needs to be specified and the number of separate Monte Carlo runs and time between statistical samples defined. The acceleration noise added at each guidance cycle is set at five percent of the thrust during burns and about 1/2 of the worst case gravity errors during coasts and can be changed if desired.

Name	Symbol	Definition	Default Value
DELS(1)	Δt_b	Time between guidance updates during burns (sec)	20 sec
DELS(2)	Δt_c	Time between guidance updates during coasts (sec)	100 sec
NOISON	-	0 - no noise 1 - navigation and acceleration perturbations	0
SIGMAR(1),SIGMAV(1)	δ_R, δ_V	Standard deviation of position and velocity navigation errors (km/sec^2) at end of second from last burn (only used during 3-burn mission)	0
SIGMAR(2),SIGMAV(2)	δ_R, δ_V	Same at end of next to last burn (km/sec^2)	0
SIGMAR(3),SIGMAV(3)	δ_R, δ_V	Same in the middle of last burn (km/sec^2)	0
PERT(1)	δ_a	Standard deviation of acceleration errors during burns (added each guidance cycle) (km/sec^2)	$\frac{.05*T}{m_0}$
PERT(2)	δ_a	Standard deviation of acceleration errors during coasts (km/sec^2)	$\frac{.5\ 10^{-4}}{R_e^2}$

Name	Symbol	Definition	Default Value
MCARLO	-	Number of Monte Carlo cases to be run	1
PTB	-	Time between output samples during burns (sec)	10 sec
PTC	-	Time between output samples during coasts (sec)	100 sec

D. General Parameters

Included here are the remainder of the parameters which may be set by NAMELIST NAMLS1 input.

Name	Symbol	Definition	Default Value
NOTARG	-	-1 Targetting only 0 Targetting and feedback guidance 1 Guidance only using parameters read in by NAMELIST NAMLS2	0
NAVOFF	-	0 Convergence status printed whenever output sample is taken in guidance mode 1 No print	1
IOUTPT	-	Integer parameter defining output device	6
EERROR	δ_e	If eccentricity less than EERROR it is set equal to 0	.01
TERROR	δ_t	If tug or target within this time (sec) tolerance of node or some mean anomaly, considered at node or mean anomaly	10. sec
RERROR	δ_i	Differences in angles (relative inclination, etc) less than this tolerance will be ignored	.5 degrees
OBLATE	-	Weighting factor used in setting oblateness effects (subroutine PERTO)	0.0

Name	Symbol	Definition	Default Value
AXIS(I)	-	Axis of rotation of the earth, must be set in relation to coordinate system chosen by targetting when oblateness is activated	$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$

E. NOTARG=1, Guidance Only Parameters (NAMSL2)

The following parameters will define the orbits of the target and vehicle, initial mass of the vehicle (all other vehicle parameters are set by NAMSL1 or default options), the initial costate vector and times array needed to define the burn and coast arcs.

Name	Symbol	Definition	Default Value
NBURNS	-	Number of burn arcs	2
XØ, TØ	$\bar{x}_0$, t_0	State of vehicle at start of mission. t_0 time at start of mission.	TØ=-1
XT, TT	$\bar{x}_T$, t_T	State of target at time t_T	TT=-1
QØ	$\bar{q}_0$	Initial costate	None
AMØ	m_0	Initial mass	28803.1155 kg (63500 lbs)
TIMES	-	Array of times defining start and end of coast and burn arcs	None

Program Output

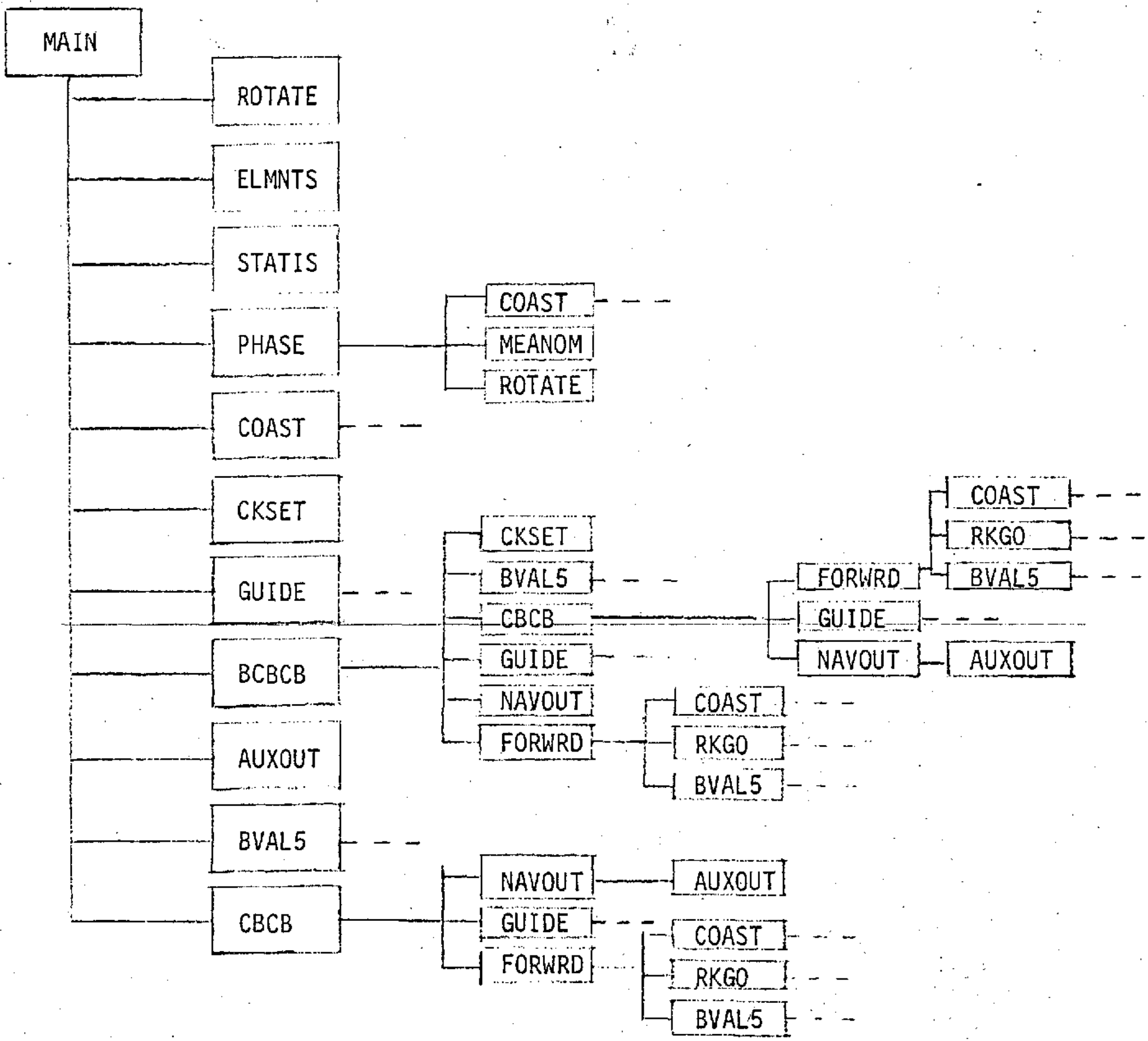
The exact program output varies with the setting of the output control parameters NAVOFF, PTB, and PTC. The nature of the output, by subroutine, is as follows:

- MAIN - error messages, impulsive approximation summary, program notes of convergence status, and converged targetting summary
- AUXOUT - summary of current convergence status
- PHASE - error messages, orbit-type message (e.g. 'CIRCULAR/CIRCULAR INCLINED ORBITS'), coast messages (when states must be advanced until proper phasing exists), and phasing-orbit messages (including relative geometry, "desired" phasing orbit, and allowable phasing orbit)
- GLMNTS - orbital elements and designation as to whether they are representative of state at start or end of a burn.
- STATIS - Monte Carlo summary
- USTAT - state, costate, and magnitude of costate vectors

PTB and PTC control the sample collection times in guidance mode, and NAVOFF controls the shutoff of the convergence-status summary (from AUXOUT) during guidance mode. In addition, there exists an internal program variable, IPRINT, which when set to 1 produces voluminous output on each call to GUIDE, detailing state and costate at predefined times on each coast arc and orbital elements at the beginning and end of each burn. Because it gives so much output, and is unlikely to be needed over an entire run, IPRINT must be set within the program.

Interdependence of Subroutines

Note that the dashed lines indicate further calls which are adequately described in the GUIDE document (except for the addition, in subroutine GUIDE, of calls to UCOAST and GLMNTS for output purposes).



Subroutine MAIN

A. Purpose

The MAIN routine controls the overall operation of the targetting and guidance program. It has four major sections. The input section, which reads the input data described in a previous section and calculates the orbital and vehicle parameters needed to perform the targetting and guidance; the phasing and impulsive-initialization section which determines the number of burn arcs, rotates the target orbit into the vehicle orbit plane and calculates the planar impulsive solution for the orbit transfer; the convergence section which first converges from the planar impulsive solution to a finite burn solution and then repeatedly reconverges with the target orbit plane rotated in ten degree steps until the desired relative inclination is obtained; the feedback guidance section which exercises the GUIDE algorithm in a realtime guidance environment, continually reconverging in the presence of perturbations and collecting Monte Carlo statistics on the performance of the algorithm.

B. Major Parameters (Input parameters discussed in Section 3)

Name	Symbol	Definition
HPGØ, HPGT, HPGX	-	Height at perigee for vehicle, target and transfer orbits (km.)
HAPØ, HAPT, HAPX	-	Height at apogee for vehicle, target and transfer orbits (km.)
AØ, AT, AX, AP	a	Semi-major axis for vehicle, target, transfer and phasing orbits (km.)
EØ, ET, EX, EP	e	Eccentricity for respective orbits
VAPØ, VAPT, VAPX, VAPP	$\lvert\bar{V}_a\rvert$	Velocity magnitude at apogee for respective orbits (km/sec)
VPGØ, VPGT, VPGX, VPGP	$\lvert\bar{V}_p\rvert$	Velocity magnitude at perigee for respective orbits (km/sec)
TAUØ, TAUT, TAUX, TAUP	τ	Period for respective orbits (sec.)
IBOUND	-	0 - Outbound mission 1 - Inbound mission

C. Method of Computation

After reading the data (as previously discussed), the routine determines whether a two burn mission will be sufficient. If the position and velocity vectors and the mean and true anomalies are not given, the true anomalies are arbitrarily chosen such that a two burn mission is possible. This is accomplished by choosing the vehicle state, for TØ=2000 seconds, at a node (perigee for outbound and apogee for inbound) in a coordinate system where perigee is in the x_1 direction. This forces the first burn to be centered at 2000 seconds and by choosing the target state at its opposite node (apogee for outbound and perigee for inbound) at TT=2000 + TAUX/2.0 (TAUX is period of desired transfer orbit) a two burn transfer is possible. For all other mission definitions the PHASE routine is called and it determines whether two burns will be sufficient and returns the state vectors defined at the time when the first burn is to begin.

Impulsive Initialization

An impulsive approximation is used as an initial guess for converging to the desired finite burn solutions. It is assumed that the optimal orbit transfer always has a burn centered about the greater apogee and this implies that the transfer orbit has as apogee the larger of the two apogees and as perigee the perigee of the other orbit. By calculating the velocities at apogee and perigee along the transfer orbit, the Δv's required are easily determined. By converting these Δv's to finite burn times, while assuming that the burns are centered at the respective nodes, and starting the mission 2000 seconds before the node, a reasonable time history for a coast-burn-coast-burn mission is defined. A reasonable estimate of initial costate $\bar{q}_0$ is also needed in order to converge the GUIDE algorithm. By investigating the impulsive case, it is determined that the direction of thrust at the node is parallel to the velocity vector and that the rate of change of thrust direction is anti-parallel to the radius vector. (The reverse directions when decreasing velocity is required, on inbound missions.) By noting that the $|\bar{q}_0|$ is arbitrary for the boundary value problem only one parameter was left to be determined, the relationship between the $|\bar{u}|$ and $|\dot{\bar{u}}|$. (Note: $\bar{q}_0^T = (\bar{u}^T, \dot{\bar{u}}^T)$.) Using the fact that the variations in $\bar{r}, \bar{v}$ form the same class of solutions as $\bar{u}, \dot{\bar{u}}$, and applying the switching condition that $|\bar{u}|$ at perigee must equal the $|\bar{u}|$ at apogee, it was found that the impulsive solution for $\bar{u}$ and $\dot{\bar{u}}$ at apogee and perigee becomes

$$\bar{u} = \bar{v}\left(\frac{r_a}{v_p} + \frac{r}{v}\right)$$

$$\dot{\bar{u}} = -\bar{r}\left(\frac{\mu}{r^3} \cdot \frac{r_a}{v_p} + \frac{v}{r}\right)$$

where r_a, v_a are the magnitudes of the position and velocity vectors at apogee; r_p, v_p are position and velocity magnitudes at perigee, and r, v are position and velocity magnitudes at either apogee or perigee (depending on where q_0 is desired) along the transfer orbit. In the program these formulas are further reduced and the $|\bar{u}|$ is chosen to be unit magnitude. The formulas become

$$\bar{u} = \frac{\bar{v}}{v}$$

$$\dot{\bar{u}} = -\bar{r} \cdot \text{FACTOR}$$

where for perigee the factor becomes

$$\text{FACTOR} = \frac{(1 + e_x/2 - e_x^2/2)\mu}{r_p^3 v_p}$$

e_x - transfer orbit eccentricity

and at apogee it is

$$\text{FACTOR} = \frac{\mu + v_a v_p r_a}{r_a^3 (v_p + v_a)}$$

When the mission is inbound and velocity needs to be reduced, the sign on both $\bar{u}$ and $\dot{\bar{u}}$ is reversed. Since this $\bar{q}_0$ is defined for the impulsive case it is good at the node and needs to be propagated back to TØ, the chosen starting time for the mission. The two burn approximate solution is now completed and the program easily converges from this to the true solution.

The approximate solution for the three burn mission is identical to that of the two burn one, except for insertion of a phasing orbit of period TAUP. For the approximate solution the phasing orbit is assumed to have the same perigee as the transfer orbit and the vehicle orbit (outbound) or target orbit (inbound). This implies that the burn at perigee is split into two burns and TAUP is chosen in subroutine PHASE to allow these burns to be as

nearly equal as possible. The typical inbound mission approximate solution thus consists of an initial burn centered at apogee of the vehicle orbit, a coast from apogee to perigee along the transfer orbit, a second burn centered about perigee of the transfer orbit, a second coast of the orbital period (perigee to perigee) along the phasing orbit and a final burn centered again at perigee. The costate vector for the inbound 3 burn planar mission (plane change is added after initial convergence) was initialized using the same formulas as the two burn case and the inbound mission successfully converges.

For the three burn outbound mission, convergence proved to be more difficult. It was discovered that the switching condition along the transfer orbit coast was very sensitive, and that the peaking characteristic of $|\bar{u}|$ at apogee and perigee was impossible to maintain when the phasing orbit was encountered before the transfer orbit. It was found that by solving the mission backwards and integrating over the transfer orbit first, reasonable convergence was attained. In order to run the GUIDE algorithm backwards from apogee on the target orbit to perigee on the vehicle orbit with increasing mass it was necessary to make the orbits retrograde by changing the sign of their velocity vectors, to change the mass rate from positive to negative, to change the sign on initial $\dot{u}$, to reduce initial mass and to alter the TIMES array. The TIMES array for the backwards three burn outbound mission is initially targetting by choosing it to be

TIMES(1) = TØ
TIMES(2) = TIMES(1) + BURN3
TIMES(3) = TIMES(2) + TAUX/2 - *BURN3 + BURN2)/2.0
TIMES(4) = TIMES(3) + BURN2
TIMES(5) = TIMES(4) + TAUP - (BURN2 + BURN1)/2.0
TIMES(6) = TIMES(5) + BURN1

Where BURN1 is the length of the burn at perigee of the vebicle orbit, BURN2 is the length of the burn at perigee of the phasing orbit, BURN3 is the length of the burn at apogee of the transfer orbit and TAUX and TAUP are the periods of the transfer and phasing orbits respectively. The initial mass is reduced to

$$m_0 = m_0 - \dot{m}(\text{BURN1} + \text{BURN2} + \text{BURN3})$$

where m is positive. QØ is initialized at apogee of the transfer orbit and then the last three components are changed in sign ($\dot{u}$).

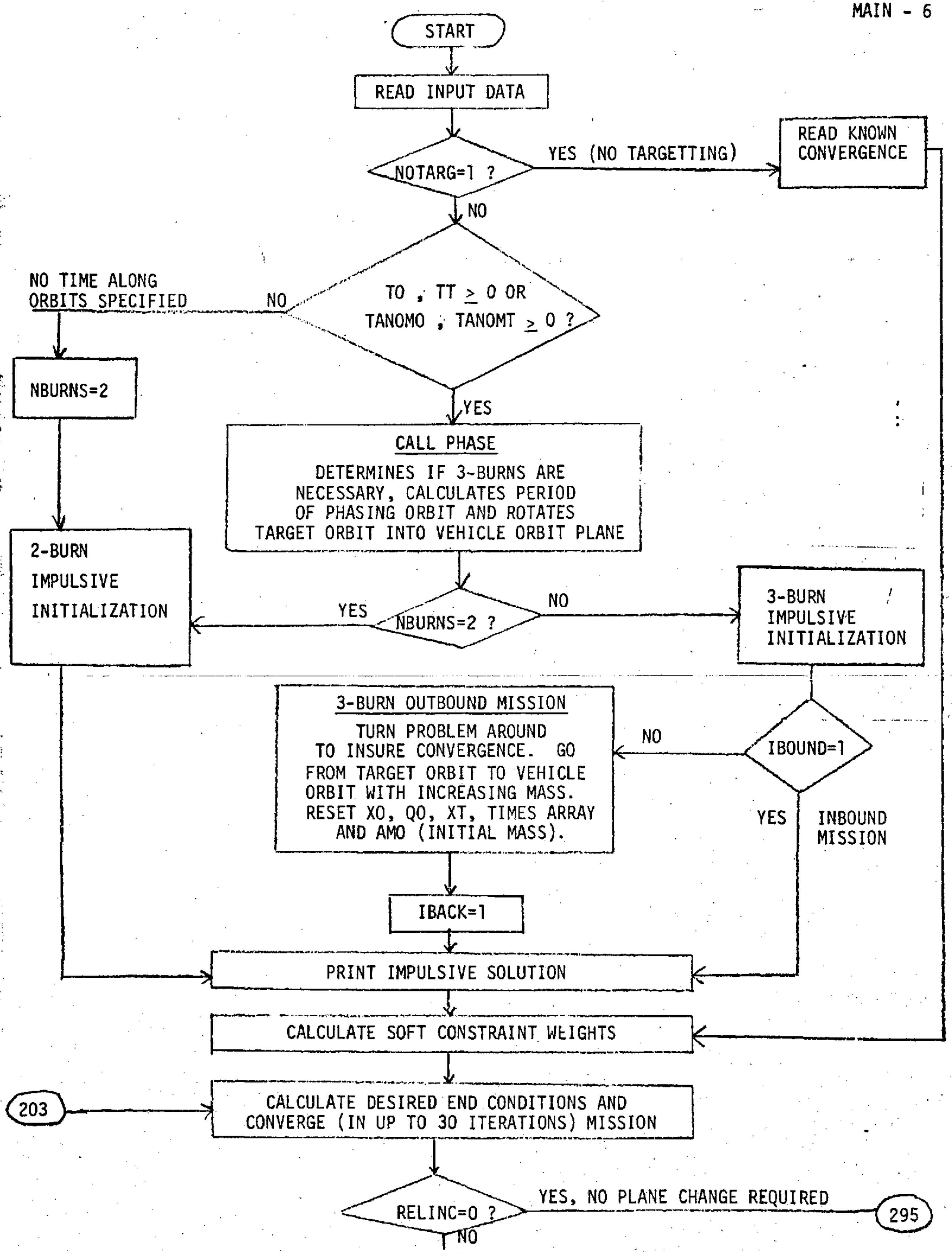
Mission Convergence

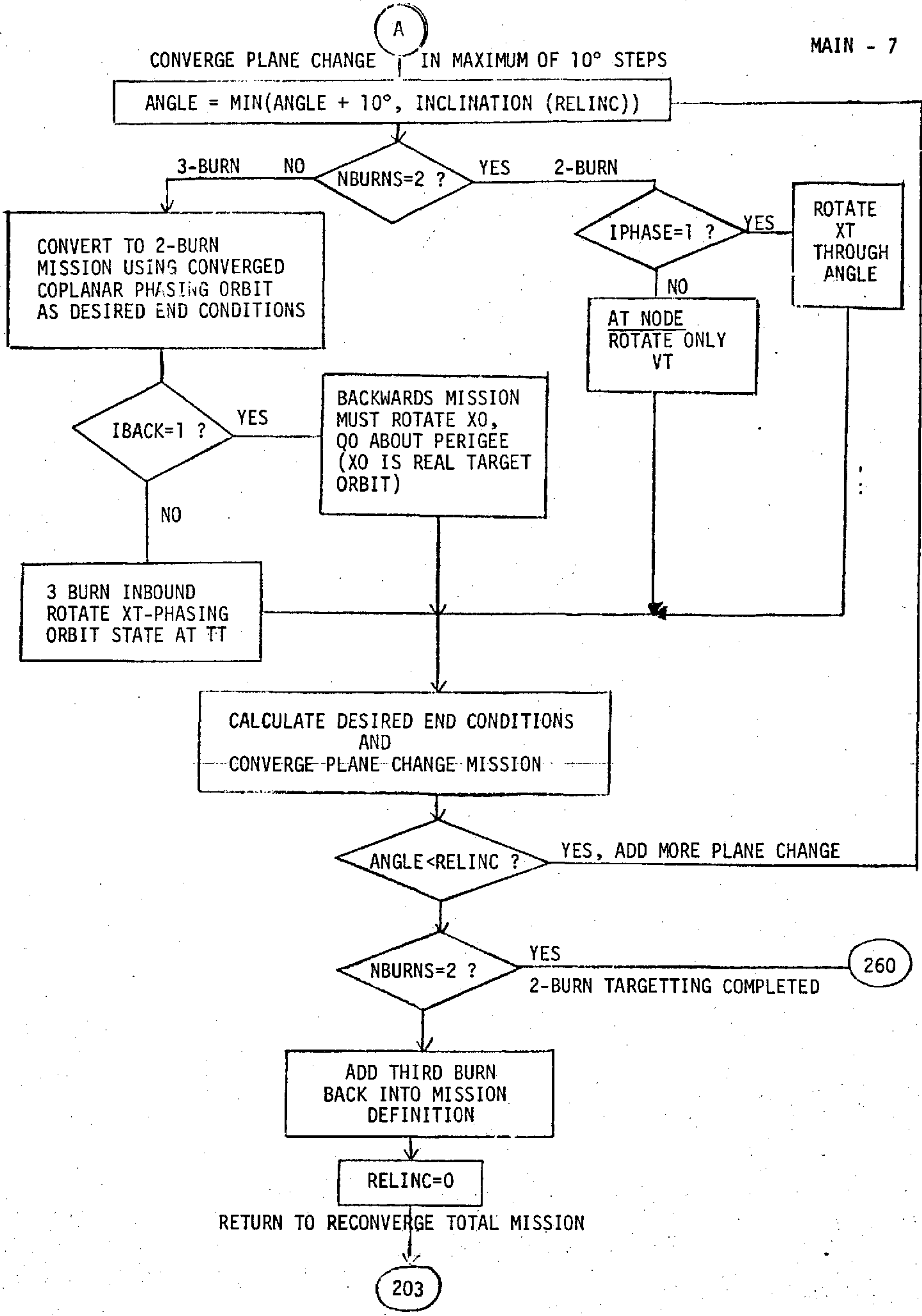
Using these approximate solutions for the two and three burn missions, the planar missions are converged in less than twenty iterations. At this point the relative inclination, RELINC, between the target and vehicle orbits is tested and if it exceeds some minimum value, the mission is altered to include the desired plane change. The target orbit is rotated in maximum of 10° steps from the vehicle orbital plane, and is reconverged at each step in the process. The two burn missions converged readily using this procedure but it was necessary to alter the three burn missions to two burn ones to obtain good convergence properties. This was accomplished by replacing the lowest orbit (target orbit for inbound and vehicle orbit for outbound) by the phasing orbit found during the planar mission convergence. The inbound mission is converged as a two burn one with the desired end conditions being the phasing orbit rotated about perigee. The outbound mission is converged backwards rotating at each step the target orbit as well as initial costate and converging to the phasing orbit. After inclusion of the total desired angular rotation, the third burn is again introduced into the mission definition and convergence for the three burn mission is attained. The outbound 3-burn mission is then turned around and solved in a forwards fashion using the final costate as initial costate and the burn and coast times derived from the backwards convergence.

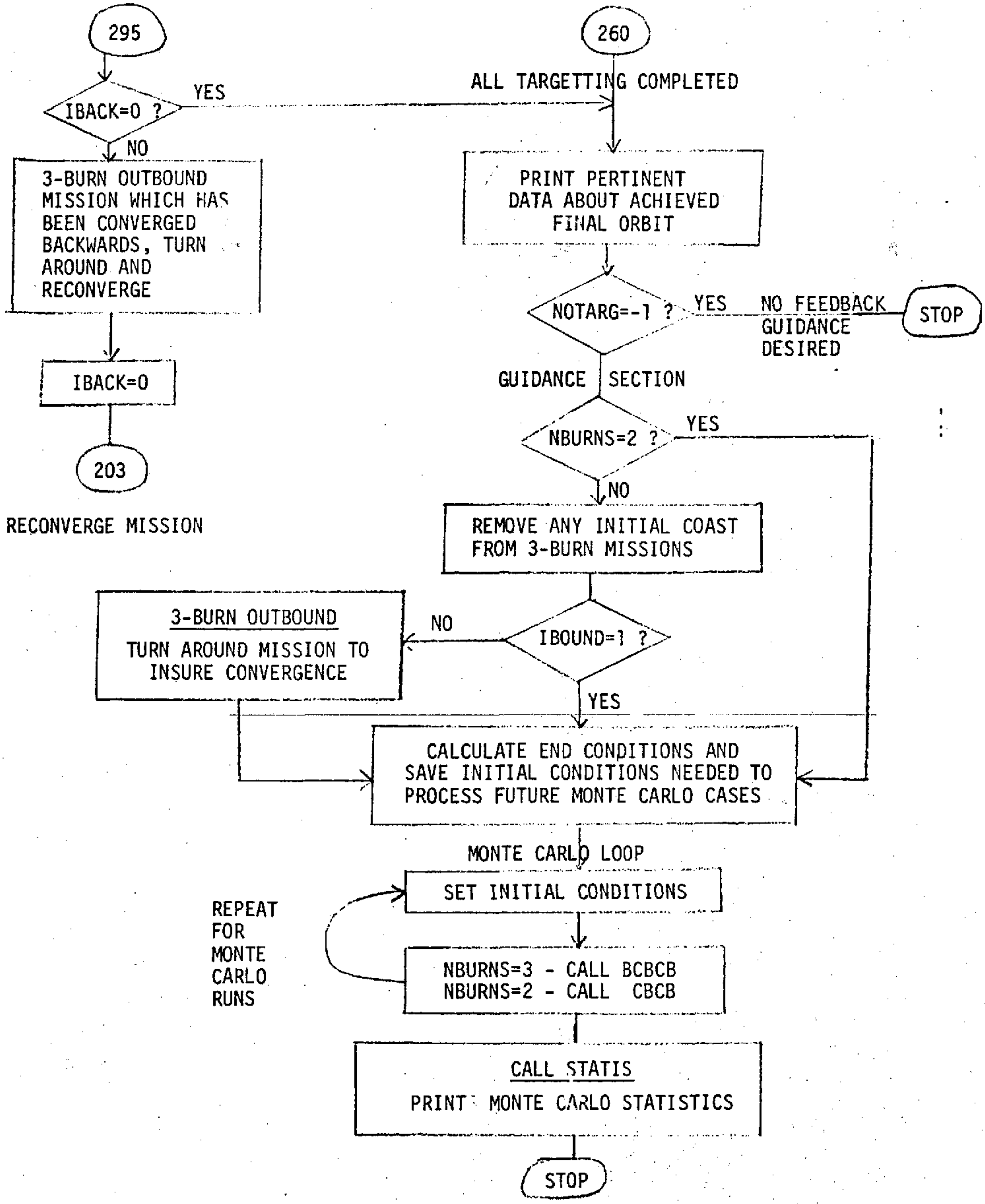
Feedback Guidance

At this point targetting is completed and a converged solution exists for guiding the vehicle into the target orbit. In the MAIN routine the major guidance function performed is to control the collection of and print the Monte Carlo statistics generated when doing feedback guidance. The routines BCBCB and CBCB called by MAIN add perturbations into the state of the vehicle and move step by step in time through a full feedback guidance cycle. At several points along each burn and coast arc, error statistics are gathered and an estimate is made of the error in meeting desired end conditions. These statistics are collected over MCARLO separate orbital transfers and a summary printout is obtained from routine STATIS.

This completes the description of the MAIN routine. A math flowchart of it is contained on the next three pages.


START
READ INPUT DATA
NOTARG=1 ?
YES (NO TARGETTING)
READ KNOWN CONVERGENCE
NO
TO , TT ≥ 0 OR TANOMO , TANOMT ≥ 0 ?
NO
NO TIME ALONG ORBITS SPECIFIED
NBURNS=2
YES
CALL PHASE
DETERMINES IF 3-BURNS ARE NECESSARY, CALCULATES PERIOD OF PHASING ORBIT AND ROTATES TARGET ORBIT INTO VEHICLE ORBIT PLANE
2-BURN IMPULSIVE INITIALIZATION
YES
NBURNS=2 ?
NO
3-BURN IMPULSIVE INITIALIZATION
3-BURN OUTBOUND MISSION
TURN PROBLEM AROUND TO INSURE CONVERGENCE. GO FROM TARGET ORBIT TO VEHICLE ORBIT WITH INCREASING MASS. RESET XO, QO, XT, TIMES ARRAY AND AMO (INITIAL MASS).
NO
IBOUND=1
YES
INBOUND MISSION
IBACK=1
PRINT IMPULSIVE SOLUTION
CALCULATE SOFT CONSTRAINT WEIGHTS
203
CALCULATE DESIRED END CONDITIONS AND CONVERGE (IN UP TO 30 ITERATIONS) MISSION
RELINC=0 ?
YES, NO PLANE CHANGE REQUIRED
295
NO


A
CONVERGE PLANE CHANGE IN MAXIMUM OF 10° STEPS
ANGLE = MIN(ANGLE + 10°, INCLINATION (RELINC))
NBURNS=2 ?
3-BURN
NO
YES
2-BURN
CONVERT TO 2-BURN MISSION USING CONVERGED COPLANAR PHASING ORBIT AS DESIRED END CONDITIONS
IPHASE=1 ?
YES
ROTATE XT THROUGH ANGLE
NO
AT NODE ROTATE ONLY VT
IBACK=1 ?
YES
BACKWARDS MISSION MUST ROTATE XO, QO ABOUT PERIGEE (XO IS REAL TARGET ORBIT)
NO
3 BURN INBOUND ROTATE XT-PHASING ORBIT STATE AT TT
CALCULATE DESIRED END CONDITIONS AND CONVERGE PLANE CHANGE MISSION
ANGLE<RELINC ?
YES, ADD MORE PLANE CHANGE
NBURNS=2 ?
YES
2-BURN TARGETTING COMPLETED
260
ADD THIRD BURN BACK INTO MISSION DEFINITION
RELINC=0
RETURN TO RECONVERGE TOTAL MISSION
203


295
IBACK=0 ?
YES
NO
3-BURN OUTBOUND MISSION WHICH HAS BEEN CONVERGED BACKWARDS, TURN AROUND AND RECONVERGE
IBACK=0
203
RECONVERGE MISSION
260
ALL TARGETTING COMPLETED
PRINT PERTINENT DATA ABOUT ACHIEVED FINAL ORBIT
NOTARG=-1 ?
YES
NO FEEDBACK GUIDANCE DESIRED
STOP
GUIDANCE SECTION
NBURNS=2 ?
YES
NO
REMOVE ANY INITIAL COAST FROM 3-BURN MISSIONS
IBOUND=1 ?
NO
YES
3-BURN OUTBOUND
TURN AROUND MISSION TO INSURE CONVERGENCE
CALCULATE END CONDITIONS AND SAVE INITIAL CONDITIONS NEEDED TO PROCESS FUTURE MONTE CARLO CASES
MONTE CARLO LOOP
SET INITIAL CONDITIONS
REPEAT FOR MONTE CARLO RUNS
NBURNS=3 - CALL BCBCB
NBURNS=2 - CALL CBCB
CALL STATIS
PRINT MONTE CARLO STATISTICS
STOP

```
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /BVLOUT/XTF(6),DELTC(6)
      COMMON /CSTAT/DATAM(5,16,9),DATAV(5,16,2)
      COMMON /ONLINE/IPRINT
      COMMON /CAIN/AXIS(3)
      COMMON /CPERT/PERT(3),DTYPE(2)
      COMMON /ODEVIC/IOUTPT
C NOTE: THE NEXT COMMON BLOCK INTERFACES ONLY WITH THE IBM RANDOM NUMBER
C GENERATOR. AN ALTERNATE METHOD IS REQUIRED TO INPUT A SEED TO THE
C GENERATOR USED IN SUBROUTINE FORWRD TO ADD RANDOM ACCELERATION NOISE
C TO THE TUG AND (OPTIONALLY) TARGET STATE.
      COMMON /RN1/NRDARL(4)
      COMMON /GIDIN/XT(6),TT,X0(6),T0,AM0,VEH(10,7),Q0(6),TIMES(6),CC(6)
      COMMON /GIDOUT/DQ0(6),DTIMES(6),DUM1(12,13),X(6),Q(6),DUM2(12,12),
     1       DD(6),DUM3(5)
      COMMON /CINDEX/IDUM4,IDUM5,IDUM6,IDUM7,IDUM8,IDUM9,ND,NDP,IDUM10
      COMMON /CMODE/MODE,IDUM11,IDUM12
      COMMON /CCK/CK
      COMMON /CWT/WEIGHT(6)
      COMMON /CPHYS/UK,REARTH,DUM13,DUM14,DUM15,OBLATE
      COMMON /CACCEL/VERROR(3),NOISON
      COMMON /UPDATE/IUPDAT
      COMMON /CNAV/TRUEMS,INAV,TACCUM,IOUT
      COMMON /OFFNAV/NAVOFF
      COMMON /CLOCK/TCLOCK
      COMMON /CPHASE/ A0,E0,AT,ET,RELINC,TAUX,PRGEE(3),HTUG( )
      COMMON /ERRORE/LERROR,TERROR,RERROR
      COMMON /COCST/ QT(6)
      COMMON /NAVUP/SIGMAR(3),SIGMAV(3)
      COMMON /INTVAL/PTB,PTC
      DIMENSION R0(3),V0(3),RT(3),VT(3),PTV(12),TUGSAV(6)
      DIMENSION VEHS(7)
      DIMENSION X0S(6),Q0S(6),CCS(6),TIMESS(6),XTS(6),QTS(6)
      DIMENSION RTA(3),VTA(3),HA(3),PGA(3),BETA(3)
      DIMENSION VTSAVE(3),STATE(6)
      DIMENSION ZH(3),ZHXR(3),PHI(6,6),QDUM(6),XPHASE(6)
      NAMELIST/NAMLS1/NOTARG,T0,TT,AXIS,OBLATE,DTYPE,PERT,AM0,THRUST,
     1       AMDOT,SPFIMP,R0,V0,RT,VT,RELINC,A0,E0,AT,ET,HAP0,HPG0,HAP
     2       T,HPGT,R0MAG,V0MAG,RTMAG,VTMAG,FLT0,FLTT,IOUTPT,LERROR,
     3       TERROR,RERROR,SIGMAR,SIGMAV,PTB,PTC,MCARLO,TANOM0,TANOMT,
     4       NAVOFF,NOISON
      NAMELIST/NAMLS2/X0,XT,Q0,TIMES,AM0,T0,TT,NBURNS,IBOUND
C * * * * * INITIALIZE CONSTANTS * * * * *
C
C PI
      PI=3.14159267
C DEGREE TO RADIAN CONVERSION
      DEGCON=PI/180.
C RADIUS OF EARTH IN KM.
      REARTH=6378.165
C EARTH GRAVITATIONAL CONSTANT IN KM**3/SEC**2
      UK=398601.5
C
C * * * * * INITIALIZE VEHICLE CHARACTERISTICS * * * * *
C
```

```
C * * * * * ENGLISH TO METRIC CONVERSIONS * * * * * *
C       1 POUND THRUST= .00444822165 KILONEWTONS
C       HENCE, THRUST(KN)=THRUST(LB)*.00444822165
C
C       1 POUND MASS= .4536 KILOGRAMS
C       HENCE, MASS(KG)=MASS(LBS)*.4536
C
C       MASS RATE (KG/SEC)= THRUST(LBS)/ISP(SEC)*.4536 KG/LB
C                         =(THRUST(KN)/.00444822165)/ISP(SEC)*.4536
C                         =THRUST(KN)/ISP*101.9716
C
C
C INITIAL MASS IN KILOGRAMS
      AM0=28603.1155
C THRUST IN KILONEWTONS
      THRUST=66.7233
C SPECIFIC IMPULSE IN SEC.
      SPEIMP=440.
C MASS RATE IN KG/SEC
      AMDUT=-1.0
C
C * * * * * INITIALIZE VARIABLES * * * * *
C
C SET INPUT DEVICE
      INPUT=1
C SET OUTPUT DEVICE
      IOUTPT=6
C SET TIME TOLERANCE (SECONDS) USED TO DETERMINE IF TUG OR TARGET IS
C CLOSE ENOUGH TO A NODE OR DESIRED MEAN ANOMALY.
      TERROR=10.
C SET ANGLE TOLERANCE (DEGREES). DIFFERENCES IN ANGLES LESS THAN RERROR
C WILL BE IGNORED.
      RERROR=.5
C SET ECCENTRICITY TOLLERANCE. ORBITS WITH ECCENTRICITIES LESS THAN
C EERROR WILL BE TREATED AS CIRCULAR.
      EERROR=.01
C SET TIMES T0,TT TO -1.0 TO INDICATE A 2-BURN MISSION.
C IF THEY ARE CHANGED BY THE INPUT DATA,A 3-BURN MISSION IS ASSUMED.
      T0=-1.0
      TT=-1.0
C SET INPUT VARIABLES TO -1.0 OR 0.0 TO INDICATE THAT THEY HAVE NOT
C BEEN READ IN THROUGH NAMELST.
      CK=1.0
      A0=0.0
      E0=-1.0
      AT=0.0
      ET=-1.0
      RELINC=0.0
      HAP0=-1.0
      HPG0=-1.0
      HAPT=-1.0
      HPGT=-1.0
      FLT0=-1.0
      FLTT=-1.0
      TANOM0=-1.0
```

```
      IANUMT=-1.0
      ROMAG=-1.0
      DO 5 I=1,3
      RO(I)=0.0
      VO(I)=0.0
      RT(I)=0.0
    5 VT(I)=0.0
      NBURNS=2
      IUPDAT=0
      NOISON=0
      NAVOFF=1
      NOTARG=0
      PTB=50.
      PTC=2000.
      PRGEE(1)=1.0
      PRGEE(2)=0.0
      PRGEE(3)=0.0
      IPHASE=0
      IBACK=0
      ITURNR=0
      IBOUND=0
      IPRINT=0
C DIRECTION OF EARTH'S AXIS IN REFERENCE COORDINATE SYSTEM.
C TUG ORBIT IS CONSIDERED EQUATORIAL. TO USE STATES IN ANOTH R CARTESIAN
C SYSTEM, THE VECTO R AXIS MUST BE CHANGED TOREFLECT THE NEW SYSTEM.
      AXIS(1)=0.0
      AXIS(2)=0.0
      AXIS(3)=1.0
C OBLATENESS
      OBLATE=0.0
C STEP SIZE IN GUIDANCE MODE DURING BURN.
      DTYPE(1)=20.
C STEP SIZE IN GUIDANCE MODE DURING COAST.
      DTYPE(2)=100.
C ZERO ARRAYS
      DO 1 I=1,12
    1 PTV(I)=0.0
      DO 3 I=1,5
      DO 3 J=1,16
      DO 2 K=1,8
    2 DATAM(I,J,K)=0.0
      DATAV(I,J,1)=0.0
      DATAV(I,J,2)=0.0
    3 DATAM(I,J,9)=0.0
      DO 4 I=1,3
      PERT(I)=0.0
      SIGMAR(I)=0.0
    4 SIGMAV(I)=0.0
      MCARLO=1
C SET TRANSVERSALITY CONDITION TO ZERO.
      TV=0.0
C SET SEEDS FOR IBM-LOCAL RANDOM NUMBER GENERATOR. THE NEXT 4 LINES MAY
C BE DROPPED,ALONG WITH COMMON BLOCK RN1, SO LONG AS A RANDOM NORMAL
C (MEAN 0,VARIANCE 1) NUMBER GENERATOR IS SUBSTITUTED .
      NRNARL(1)=0
```

```
      NRNARL(2)=0
      NRNARL(3)=33737
      NRNARL(4)=0
C READ INPUT FROM NAMELIST.
      READ(INPUT,NAMLS1)
C COMPUTE MASS RATE IF NOT READ IN.
      IF(AMDOT.LT.0.0) AMDOT=THRUST/SPEIMP*101.9716
C PERTURBATIONS IN TUG ACCELERATION DURING GUIDANCE. PERT(1) IS DURING
C A BURN AND PERT(2) DURING A COAST. PERT(3) CAN BE USED FOR INTRODUCING
C PERTURBATIONS IN THE TARGET ACCELERATION.
      IF(PERT(1)+PERT(2).GT.1.D-8) GO TO 6
C
C
      PERT(1)=.05*THRUST/AMO
      PERT(2)=.00005*UK/(6700.**2)
    6 CONTINUE
C STORE VEHICLE SPECS IN WORKING ARRAY VEH(10,7)
      VEH(1,1)=AMO
      VEH(1,2)=AMDOT
      VEH(1,3)=10000.
      VEH(1,4)=THRUST
      VEH(1,5)=0.0
      VEH(1,6)=0.0
      VEH(1,7)=25.0
C CHECK IF GUIDANCE IS DESIRED WITHOUT PRELIMINARY TARGETTING (OPTIMAL
C SOLUTION ALREADY EXISTS). A PASS WILL BE MADE THROUGH A CONVERGENCE
C LOOP TO INSURE THAT THE SOLUTION IS VALID.
      IF(NOTARG.EQ.1) ITURNR=3
      IF(NOTARG.EQ.1) GO TO 204
C TARGETTING IS DESIRED.DETERMINE IF A 2 OR 3-BURN MISSION IS DESIRED.
C IF TO.GE.0.0 AND TT.GE.0.0,A 3-BURN IS ASSUMED.
C ALSO, IF THE TRUE ANOMALIES WERE,SPECIFIED, A 3-BURN MISSION IS
C ASSUMED.
      IF(TO.GE.0.0.AND.TT.GE.0.0) GO TO 500
      IF(TANOMO.GE.0.0.AND.TANOMT.GE.0.0) GO TO 500
C
C * * * * * TWO-BURN MISSION SPECIFIED * * * * *
C
C CHECK FOR COMPLETE SET OF 2-BURN INPUT DATA.
C CHECK FIRST WHETHER SEMI-MAJOR AXIS (AO) AND
C ECCENTRICITY (EO) OF INITIAL ORBIT AND (AT,ET) OF FINAL
C ORBIT WERE SPECIFIED.
      IF(AO.GT.REARTH.AND.EO.GE.0.0.AND.AT.GT.REARTH.AND.ET.
     1GE.0.0) GO TO 100
C (AO,EO) AND (AT,ET) WERE NOT SPECIFIED. CHECK IF HEIGHTS AT APOGEE
C AND PERIGEE (KM) WERE SPECIFIED.
      IF(HAPO.GE.0.0.AND.HPGO.GE.0.0.AND.HAPT.GE.0.0.AND.HPGT.GE.0.0)
     1 GO TO 50
C HEIGHTS NOT SPECIFIED. CHECK IF MAGNITUDES OF POSITION AND VELOCITY
C AND FLIGHT ANGLES WERE SPECIFIED.
C THE FLIGHT ANGLE IS DEFINED AS THE ANGLE BETWEEN THE POSITION AND
C VELOCITY VECTOR, MEASURED IN THE DIRECTION OF ORBITAL MOTION. IT IS
C .LE.90 DEGREES BETWEEN PERIGEE AND APOGEE AND .GE.90 DEGREES BETWEEN
C APOGEE AND PERIGEE, AND DETERMINES THE ANGULAR MOMENTUM THROUGH RMAG*
C VMAG*DABS(SIN(FLT))
```

```
      IF(ROMAG.GE.REARTH.AND.VOMAG.GE..1.AND.FLTO.GE.0.0.AND.              MAI00221
     1   RTMAG.GE.REARTH.AND.VTMAG.GE..1.AND.FLTT.GE.0.0) GO TO 75         MAI00222
C ALLOWABLE SET OF DATA FOR 2-BURN MISSION WAS NOT READ IN.  DUMP           MAI00223
C VARIABLES AND TERMINATE.                                                  MAI00224
      WRITE(IOUTPT,3000)                                                    MAI00225
3000  FORMAT(' EXECUTION TERMINATING. IMPROPER DATA SPECIFIED FOR 2-BURNMAI00226
     1 MISSION')                                                            MAI00227
      WRITE(IOUTPT,3001) A0,E0,AT,ET,RELINC                                 MAI00228
3001  FORMAT(' SET 1 -',/,' A0=',D14.6,' E0=',D14.6,' AT=',D14.6,           MAI00229
     1' ET=',D14.6,' RELINC=',D14.6,//)                                     MAI00230
25    WRITE(IOUTPT,3002) HAP0,HPG0,HAPT,HPGT,RELINC                         MAI00231
3002  FORMAT(' SET 2 -',/,' HAP0=',D14.6,' HPG0=',D14.6,' HAPT=',D14.6,'MAI00232
     1 HPGT=',D14.6,' RELINC=',D14.6,//)                                    MAI00233
      WRITE(IOUTPT,3003) ROMAG,VOMAG,FLTO,RTMAG,VTMAG,FLTT,RELINC           MAI00234
3003  FORMAT(' SET 3 -',/,' ROMAG=',D14.6,' VOMAG=',D14.6,' FLTO=',D14.6MAI00235
     1,/,' RTMAG=',D14.6,' VTMAG=',D14.6,' FLTT=',D14.6,' RELINC=',D14.6MAI00236
     1,//)                                                                  MAI00237
      STOP                                                                  MAI00238
C CONVERT HEIGHTS AT APOGEE AND PERIGEE INTO ORBITAL ELEMENTS.              MAI00239
50    A0=REARTH+(HAP0+HPG0)/2.                                              MAI00240
      E0=(HAP0+REARTH)/A0-1.0                                               MAI00241
      IF(E0.LT.EERROR) E0=0.0                                               MAI00242
      AT=REARTH+(HAPT+HPGT)/2.                                              MAI00243
      ET=(HAPT+REARTH)/AT-1.0                                               MAI00244
      IF(ET.LT.EERROR) ET=0.0                                               MAI00245
      GO TO 110                                                             MAI00246
C CONVERT POSITION,VELOCITY AND FLIGHT ANGLES TO ORBITAL ELEMENTS.          MAI00247
75    A0=UK*ROMAG/(2.*UK-VOMAG**2*ROMAG)                                    MAI00248
      HOMAG=DABS(ROMAG*VOMAG*DSIN(FLTO*DEGCON))                             MAI00249
      E0=DSQRT(1.-HOMAG**2/(A0*UK))                                         MAI00250
      IF(E0.LT.EERROR) E0=0.0                                               MAI00251
      AT=UK*RTMAG/(2.*UK-VTMAG**2*RTMAG)                                    MAI00252
      HTMAG=DABS(RTMAG*VTMAG*DSIN(FLTT*DEGCON))                             MAI00253
      ET=DSQRT(1.-HTMAG**2/(AT*UK))                                         MAI00254
      IF(ET.LT.EERROR) ET=0.0                                               MAI00255
100   HAP0=A0*(1.+E0)-REARTH                                                MAI00256
      HPG0=A0*(1.-E0)-REARTH                                                MAI00257
      HAPT=AT*(1.+ET)-REARTH                                                MAI00258
      HPGT=AT*(1.-ET)-REARTH                                                MAI00259
      IF(HPG0.GT.0.0.AND.HPGT.GT.0.0) GO TO 110                             MAI00260
      WRITE(IOUTPT,3150) HPG0,HPGT                                          MAI00261
3150  FORMAT(' EXECUTION TERMINATING.',/,' HEIGHT AT PERIGEE OF TUG='       MAI00262
     1,D14.6,/,' HEIGHT AT PERIGEE OF TARGET=',D14.6)                       MAI00263
      STOP                                                                  MAI00264
C                                                                           MAI00265
C DETERMINE INITIAL AND FINAL RADII. TEST FIRST IF AN INBOUND OR            MAI00266
C OUTBOUND MISSION. IF APOGEE OF THE TARGET ORBIT IS LESS THAN APOGEE       MAI00267
C OF THE TUG ORBIT, AN INBOUND MISSION IS ASSUMED.                          MAI00268
C CHECK FOR GREATER APOGEE.                                                 MAI00269
110   IF(HAP0.GT.HAPT) GO TO 125                                            MAI00270
C OUTBOUND MISSION (FROM PERIGEE TO APOGEE).                                MAI00271
      IBOUND=0                                                              MAI00272
      RI=REARTH+HPG0                                                        MAI00273
      RF=REARTH+HAPT                                                        MAI00274
      GO TO 150                                                             MAI00275
```

```
C INBOUND MISSION (FROM APOGEE TO PERIGEE).                                  MAI00276
  125   IBOUND=1                                                              MAI00277
        RI=REARTH+HAPO                                                        MAI00278
        RF=REARTH+HPGT                                                        MAI00279
C CALCULATE INITIAL AND FINAL VELOCITIES.                                     MAI00280
  150   VI=DSQRT(UK*(2./RI-1./AO))                                            MAI00281
        VF=DSQRT(UK*(2./RF-1./AT))                                            MAI00282
C DETERMINE VELOCITIES AT END POINTS OF TUG,TARGET ORBITS                     MAI00283
        VAPO=DSQRT(UK*(2./(HAPO+REARTH)-1./AO))                               MAI00284
        VPGO=DSQRT(UK*(2./(HPGO+REARTH)-1./AO))                               MAI00285
        VAPT=DSQRT(UK*(2./(HAPT+REARTH)-1./AT))                               MAI00286
        VPGT=DSQRT(UK*(2./(HPGT+REARTH)-1./AT))                               MAI00287
C DEFINE STATES IN ARBITRARY COORDINATE SYSTEM. UNLESS ONE IS IMPLIED         MAI00288
C THROUGH THE STATE VECTORS (PHASE WAS CALLED). THE TUG WILL BE LOCATED       MAI00289
C AT PERIGEE/APOGEE AND THE TARGET AT APOGEE/PERIGEE DEPENDING ON THE         MAI00290
C VALUE OF IBOUND. PERIGEE IS IN THE DIRECTION (1,0,0) AND THE H              MAI00291
C VECTOR IN THE DIRECTION (0,0,1)                                             MAI00292
C IF IPHASE=1, THE STATE VECTORS WERE SET UP IN SUBROUTINE PHASE.             MAI00293
        IF(IPHASE.EQ.1) GO TO 175                                             MAI00294
        SIGN=1.0                                                              MAI00295
        IF(IBOUND.EQ.1) SIGN=-1.0                                             MAI00296
        RO(1)=RI*SIGN                                                         MAI00297
        RO(2)=0.0                                                             MAI00298
        RO(3)=0.0                                                             MAI00299
        VO(1)=0.0                                                             MAI00300
        VO(2)=VI*SIGN                                                         MAI00301
        VO(3)=0.0                                                             MAI00302
        RT(1)=-RF*SIGN                                                        MAI00303
        RT(2)=0.0                                                             MAI00304
        RT(3)=0.0                                                             MAI00305
        VT(1)=0.0                                                             MAI00306
        VT(2)=-VF*SIGN                                                        MAI00307
        VT(3)=0.0                                                             MAI00308
C SET UP TRANSFER ORBIT                                                       MAI00309
  175   AX=(RI+RF)/2.                                                         MAI00310
        EX=DMAX1(RI,RF)/AX-1.0                                                MAI00311
        HAPX=AX*(1.+EX)-REARTH                                                MAI00312
        HPGX=AX*(1.-EX)-REARTH                                                MAI00313
        VAPX=DSQRT(UK*(2./DMAX1(RI,RF)-1./AX))                                MAI00314
        VPGX=DSQRT(UK*(2./DMIN1(RI,RF)-1./AX))                                MAI00315
C DEFINE DELTA V'S (+ OR -)                                                   MAI00316
C THE INITIAL DELTA V IS DEFINED AS THE VELOCITY AT THE APOGEE/PERIGEE        MAI00317
C ON THE TRANSFER ORBIT MINUS THE INITIAL VELOCITY. THE FINAL DELTA V ISMAI00318
C DEFINED AS THE FINAL VELOCITY MINUS THE VELOCITY AT PERIGE /APOGEE ON       MAI00319
C THE TRANSFER ELLIPSE.                                                       MAI00320
        IF(IBOUND.EQ.1) GO TO 180                                             MAI00321
        DELTVI=VPGX-VI                                                        MAI00322
        DELTVF=VF-VAPX                                                        MAI00323
        VELXFR=VPGX                                                           MAI00324
        GO TO 195                                                             MAI00325
  180   DELTVI=VAPX-VI                                                        MAI00326
        DELTVF=VF-VPGX                                                        MAI00327
        VELXFR=VAPX                                                           MAI00328
C       DETERMINE ORBITAL PERIODS.                                            MAI00329
  195   TAUO=2.*PI*DSQRT(AO**3/UK)                                            MAI00330
```

```
      TAUX=2.*PI*DSQRT(AX**3/UK)                                       MAI00331
      TAUT=2.*PI*DSQRT(AT**3/UK)                                       MAI00332
C DETERMINE FINITE BURNS FROM DESIRED DELTA V'S AND INITIAL MASS.      MAI00333
      BURN1=-AM0/AMDOT*(DEXP(-AMDOT*DABS(DELTVI)/THRUST)-1.0)          MAI00334
      DELTAM=BURN1*AMDOT                                               MAI00335
      BURN2=-(AM0-DELTAM)/AMDOT*(DEXP(-AMDOT*DABS(DELTVF)/THRUST)-1.0) MAI00336
      COAST1=TAUX/2.-(BURN1+BURN2)/2.                                  MAI00337
C SET UP TIMES ARRAY FOR 2-BURN MISSION WITH TIME T0=0 ARBITRARILY 2000 MAI00338
C SECONDS BEFORE THE TUG IS DUE AT THE NODE, UNLESS T0 AND TT WERE     MAI00339
C ORIGINALLY SPECIFIED AS .GE. 0.0 (IPHASE=1) IN WHICH CASE T0 IS      MAI00340
C RUN BACK BY 2000 SEC, SO LONG AS IT IS .GE. 0.0 .                    MAI00341
      TIMES(1)=0.0                                                     MAI00342
      TIMES(2)=0.0                                                     MAI00343
      IF(IPHASE.EQ.0) GO TO 196                                        MAI00344
      SHIFT=0.0                                                        MAI00345
      IF(T0-2000..LT.0.0) SHIFT=2000.-T0                               MAI00346
 3050 FORMAT(' TIME SCALE CHANGED BY',F10.2,' SECONDS TO ALLOW FOR INITIMAI00347
     1AL COAST.',/,' T0 AND TT NOW EQUAL 0.0 AND',F10.2)               MAI00348
      IF(T0-2000.0.LT.0.0) TT=TT+2000.-T0                              MAI00349
      IF(T0-2000.0.LT.0.0) T0=0.0                                      MAI00350
      IF(SHIFT.GT..001) WRITE(IOUTPT,3050) SHIFT,TT                    MAI00351
      IF(T0.GT.0.001) T0=T0-2000.                                      MAI00352
      GO TO 197                                                        MAI00353
  196 TT=2000.+TAUX/2.                                                 MAI00354
      T0=0.0                                                           MAI00355
  197 TIMES(3)=2000.-BURN1/2.+T0                                       MAI00356
      TIMES(4)=TIMES(3)+BURN1                                          MAI00357
      TIMES(5)=TIMES(4)+COAST1                                         MAI00358
      TIMES(6)=TIMES(5)+BURN2                                          MAI00359
C STORE THE TUG AND TARGET STATES IN THE WORKING VARIABLES X0,XT       MAI00360
      DO 200 I=1,3                                                     MAI00361
      X0(I)=R0(I)                                                      MAI00362
      X0(I+3)=V0(I)                                                    MAI00363
      XT(I)=RT(I)                                                      MAI00364
  200 XT(I+3)=VT(I)                                                    MAI00365
C DEFINE IMPULSIVE COSTATE. U IS OF UNIT MAGNITUDE AND ALONG VELOCITY  MAI00366
C VECTOR AT THE NODE AND U-DOT IS ALONG THE EARTH RADIUS VECTOR.       MAI00367
      IF(IBOUND.EQ.1) GO TO 2005                                       MAI00368
      SIGN=1.0                                                         MAI00369
      FACTOR=(1.+EX/2.-EX**2/2.)*UK/(R1**3*VELXFR)                     MAI00370
      GO TO 2007                                                       MAI00371
 2005 SIGN=-1.0                                                        MAI00372
      FACTOR=(UK+VAPX*VPGX*(HAPX+REARTH))/((HAPX+REARTH)**3*(VAPX+VPGX))MAI00373
 2007 CONTINUE                                                         MAI00374
      DO 201 I=1,3                                                     MAI00375
      Q0(I)=X0(I+3)/V1*SIGN                                            MAI00376
  201 Q0(I+3)=-X0(I)*FACTOR*SIGN                                       MAI00377
      NO=0                                                             MAI00378
C COAST TUG BACK ARBITRARY 2000 SEC. BEFORE START OF THE 1ST BURN.     MAI00379
      CALL COAST(X0,Q0,-2000.,X0,Q0,PHI,PHI)                           MAI00380
      TBEGIN=TIMES(3)                                                  MAI00381
      GO TO 202                                                        MAI00382
C * * * * * THREE-BURN MISSION * * * * *                               MAI00383
C                                                                      MAI00384
C A 3-BURN MISSION IS ASSUMED, UNLESS THE PHASING WILL ALLOW           MAI00385
```

```
C A 2-BURN MISSION. CHECK IF THE TUG AND TARGET STATES WERE SPECIFIED.
 500    NBURNS=3
        IF(ROMAG.GT.0.0) GO TO 501
        ROMAG=DSQRT(RO(1)**2+RO(2)**2+RO(3)**2)
        VOMAG=DSQRT(VO(1)**2+VO(2)**2+VO(3)**2)
        RTMAG=DSQRT(RT(1)**2+RT(2)**2+RT(3)**2)
        VTMAG=DSQRT(VT(1)**2+VT(2)**2+VT(3)**2)
        IF(ROMAG.GT.REARTH.AND.VOMAG.GT..1.AND.
       1 RTMAG.GT.REARTH.AND.VTMAG.GT..1) GO TO 600
C STATES WERE NOT SPECIFIED. CHECK IF ORBITAL ELEMENTS WERE.
        IF(AO.GE.REARTH.AND.EO.GE.0.0.AND.
       1  AT.GE.REARTH.AND.ET.GE.0.0) GO TO 510
C CHECK IF HEIGHTS AT APOGEE AND PERIGEE (KM) WERE SPECIFIED
        IF(HAPO.GE.0.0.AND.HPGO.GE.0.0.AND.HAPT.GE.0.0
       1 .AND.HPGT.GE.0.0) GO TO 505
C       CHECK IF MAGNITUDES OF POSITION AND VELOCITY AND FLIGHT
C ANGLES WERE SPECIFIED.( FLIGHT ANGLE DEFINED IN 2-BURN COMMENTS.)
 501    IF(ROMAG.GE.REARTH.AND.VOMAG.GE..1.AND.FLTO.GE.0.0.AND.
       1RTMAG.GE.REARTH.AND.VTMAG.GE..1.AND.FLTT.GE.0.0) GO TO 507
C INADEQUATE ELEMENTS SPECIFIED. STOP.
        WRITE(IOUTPT,3100) RO,VO,RT,VT,RELINC,AO,EO,AT,ET
 3100   FORMAT(' EXECUTION TERMINATING. IMPROPER DATA SPECIFIED FOR 3-BURN
       1 MISSION.',/,' SET 1 -',/,' RO=',3D14.6,' VO=',3D14.6,/,' RT=',3D1
       24.6,' VT=',3D14.6,/,' RELINC=',D14.6,//,' SET 2 -',/,' AO=',D14.6,
       3' EO=',D14.6,' AT=',D14.6,' ET=',D14.6,/)
        GO TO 25
C
C CONVERT HEIGHTS INTO ORBITAL ELEMENTS.
 505    AO=REARTH+(HAPO+HPGO)/2.0
        EO=(HAPO+REARTH)/AO-1.0
        IF(EO.LT.ERROR) EO=0.0
        AT=REARTH+(HAPT+HPGT)/2.0
        ET=(HAPT+REARTH)/AT-1.0
        IF(ET.LT.ERROR) ET=0.0
        GO TO 510
C CONVERT POSITION,VELOCITY AND FLIGHT ANGLES INTO ORBITAL ELEMENTS.
 507    AO=UK*ROMAG/(2.*UK-VOMAG**2*ROMAG)
        HOMAG=DABS(ROMAG*VOMAG*DSIN(FLTO*DEGCON))
        EO=DSQRT(1.-HOMAG**2/(AO*UK))
        IF(EO.LT.ERROR) EO=0.0
        AT=UK*RTMAG/(2.*UK-VTMAG**2*RTMAG)
        HTMAG=DABS(RTMAG*VTMAG*DSIN(FLTT*DEGCON))
        ET=DSQRT(1.-HTMAG**2/(AT*UK))
        IF(ET.LT.ERROR) ET=0.0
 510    HPGO=AO*(1.-EO)-REARTH
        HPGT=AT*(1.-ET)-REARTH
C
C ELEMENTS WERE SPECIFIED. CHECK WHETHER TRUE ANOMALIES WERE SPECIFIED.
        IF(HPGO.GE.0.0.AND.HPGT.GE.0.0) GO TO 511
        WRITE(IOUTPT,3150) HPGO,HPGT
        STOP
 511    IF(TANOMO.GE.0.0.AND.TANOMT.GE.0.0) GO TO 550
C ANOMALIES WERE NOT SPECIFIED. THE TIMES TO,TT WILL NOW BE CONSIDERED
C THE MEAN ANOMALIES (TIMES SINCE PERIGEE OR, IF CIRCULAR, SINCE THE
C NODE)
```

```
C CHECK IF EITHER IS GREATER THAN ITS ORBITAL PERIOD.                  MAI00441
      TAU0=2.*PI*DSQRT(A0**3/UK)                                        MAI00442
      TAUT=2.*PI*DSQRT(AT**3/UK)                                        MAI00443
      IF(T0.LT.TAU0.AND.TT.LT.TAUT) GO TO 520                           MAI00444
C TIMES ARE NOT LESS THAN ONE ORBITAL PERIOD. STOP                      MAI00445
      WRITE(IOUTPT,3102) T0,TT,TAU0,TAUT                                MAI00446
 3102 FORMAT(' EXECUTION TERMINATING. MEAN ANOMALIES EXCEED ONE ORBITAL MAI00447
     1 PERIOD',/,' T0=',D14.6,' TT=',D14.6,' TAU0=',D14.6,' TAUT=',D14. MAI00448
     16)                                                                MAI00449
      STOP                                                              MAI00450
C TIMES SINCE PERIGEE PASSAGE ARE REASONABLE. DEFINE TUG AND TARGET     MAI00451
C STATES AT PERIGEE AND PROPAGATE AHEAD VIA CALLS TO COAST. TO THE      MAI00452
C TIMES GIVEN BY THE MEAN ANOMALIES.                                    MAI00453
  520 STATE(1)=A0*(1.-E0)                                               MAI00454
      STATE(2)=0.0                                                      MAI00455
      STATE(3)=0.0                                                      MAI00456
      STATE(4)=0.0                                                      MAI00457
      STATE(5)=DSQRT(UK*(2./STATE(1)-1./A0))                            MAI00458
      STATE(6)=0.0                                                      MAI00459
      N0=-1                                                             MAI00460
      CALL COAST(STATE,QDUM,T0,STATE,QDUM,PHI,PHI)                      MAI00461
      DO 525 I=1,3                                                      MAI00462
      R0(I)=STATE(I)                                                    MAI00463
  525 V0(I)=STATE(I+3)                                                  MAI00464
      STATE(1)=AT*(1.-ET)                                               MAI00465
      STATE(2)=0.0                                                      MAI00466
      STATE(3)=0.0                                                      MAI00467
      VMAGNT=DSQRT(UK*(2./STATE(1)-1./AT))                              MAI00468
      STATE(4)=0.0                                                      MAI00469
      STATE(5)=VMAGNT*DCOS(RELINC*DEGCON)                               MAI00470
      STATE(6)=VMAGNT*DSIN(RELINC*DEGCON)                               MAI00471
      CALL COAST(STATE,QDUM,TT,STATE,QDUM,PHI,PHI)                      MAI00472
      DO 530 I=1,3                                                      MAI00473
      RT(I)=STATE(I)                                                    MAI00474
  530 VT(I)=STATE(I+3)                                                  MAI00475
C SET BOTH TIMES TO 2000. AFTER PROPAGATING STATES THROUGH THEIR        MAI00476
C MEAN ANOMALIES.                                                       MAI00477
      T0=2000.                                                          MAI00478
      TT=2000.0                                                         MAI00479
      GO TO 600                                                         MAI00480
C                                                                       MAI00481
C TRUE ANOMALIES WERE SPECIFIED. SET UP COORDINATE SYSTEM SUCH          MAI00482
C THAT THE X(1) AXIS IS TOWARDS TUG PERIGEE. X(3) IS ALONG              MAI00483
C H, AND X(2) IS X(3) CROSS X(1).                                       MAI00484
  550 RMAG=A0*(1.-E0**2)/(1.+E0*DCOS(TANOM0*DEGCON))                    MAI00485
      R0(1)=RMAG*DCOS(TANOM0*DEGCON)                                    MAI00486
      R0(2)=RMAG*DSIN(TANOM0*DEGCON)                                    MAI00487
      R0(3)=0.0                                                         MAI00488
      VMAG=DSQRT(UK/(A0*(1.-E0**2)))                                    MAI00489
      V0(1)=-VMAG*DSIN(TANOM0*DEGCON)                                   MAI00490
      V0(2)=VMAG*(E0+DCOS(TANOM0*DEGCON))                               MAI00491
      V0(3)=0.0                                                         MAI00492
      RMAG=AT*(1.-ET**2)/(1.+ET*DCOS(TANOMT*DEGCON))                    MAI00493
      RT(1)=RMAG*DCOS(TANOMT*DEGCON)                                    MAI00494
      RT(2)=RMAG*DSIN(TANOMT*DEGCON)                                    MAI00495
```

```
      RT(3)=0.0
      VMAG=DSQRT(UK/(AT*(1.-ET**2)))
      VT(1)=-VMAG*DSIN(TANOMT*DEGCON)
      VT(2)=VMAG*(ET+DCOS(TANOMT*DEGCON))
      VT(3)=0.0
C ADD INCLINATION TO TARGET ORBIT.
      RT(1)=RT(1)
      RT(3)=RT(2)*DSIN(RELINC*DEGCON)
      RT(2)=RT(2)*DCOS(RELINC*DEGCON)
      VT(1)=VT(1)
      VT(3)=VT(2)*DSIN(RELINC*DEGCON)
      VT(2)=VT(2)*DCOS(RELINC*DEGCON)
      T0=2000.0
      TT=2000.0
      GO TO 700
C
C TUG AND TARGET STATES WERE SPECIFIED. CHECK IF THE TIMES ARE THE SAME.
C THE STATES MUST BE SPECIFIED AT THE SAME TIME WHEN PHASE IS CALLED.
C T0 IS ASSUMED TO BE THE REAL CLOCK TIME AND NEVER DECREASES.
 600  IF(DABS(T0-TT).LT..1) GO TO 700
C COAST TARGET STATE BACKWARDS OR FORWARDS IN TIME AS REQUIRED.
C SUBROUTINE COAST EXPECTS A 6-VECTOR OF STATES.
      DO 601 I=1,3
      STATE(I)=RT(I)
 601  STATE(I+3)=VT(I)
      NO=-1
      CALL COAST(STATE,QDUM,T0-TT,STATE,QDUM,PHI,PHI)
      TT=T0
      DO 602 I=1,3
      RT(I)=STATE(I)
 602  VT(I)=STATE(I+3)
C ARBITRARILY CHANGE TIME ORIGIN.IF T0.LT.2000., INCREASING T0 TO 2000.
C  SECONDS TO ALLOW FOR POSSIBLE INITIAL COASTS.
      IF(T0.GT.2000.) GO TO 700
      TEMP=2000.-T0
      TT=TT+TEMP
      T0=2000.
      WRITE(IOUTPT,3120) TEMP,T0,TT
 3120 FORMAT(' CHANGED TIME ORIGIN TO ALLOW FOR INITIAL COASTS.',/,
     1' INCREASED T0 AND TT BY ',F7.2,'. T0 AND TT NOW EQUAL ',
     2E16.8,' , ',E16.8)
C CALL PHASE TO DETERMINE IF RENDEZVOUS IS POSSIBLE WITH 2 OR 3
C BURNS. FOR CIRCULAR TO CIRCULAR CO-PLANAR MISSIONS,A 2-BURN
C MISSION IS ALWAYS POSSIBLE. FOR OTHER GEOMETRIES, A 2 OR 3 BURN
C MISSION MAY BE POSSIBLE. EXECUTION WILL TERMINATE IN PHASE, IF NO
C MISSION IS POSSIBLE WITHIN THE ALLOTTED TIME.
 700  CALL PHASE(R0,V0,RT,VT,NBURNS,TCOAST,TAUP)
      IPHASE=1
      TAU0=2.*PI*DSQRT(A0**3/UK)
      TAUT=2.*PI*DSQRT(AT**3/UK)
      T0=T0+TCOAST
      TT=T0
      IF(NBURNS.EQ.2) GO TO 100
C DETERMINE WHETHER AN INBOUND OR OUTBOUND MISSION IS REQUIRED, AFTER
C FORMING HEIGHTS AT APOGEE AND PERIGEE.
```

```
      HAP0=A0*(1.+E0)-REARTH
      HPG0=A0*(1.-E0)-REARTH
      HAPT=AT*(1.+ET)-REARTH
      HPGT=AT*(1.-ET)-REARTH
      IF(HAP0.GT.HAPT) IBOUND=1
      AP=(TAUP**2*UK/(4.*PI**2))**.333333333
      IF(IBOUND.EQ.0) HPGP=HPG0
      IF(IBOUND.EQ.1) HPGP=HPGT
      EP=1.0-(HPGP+REARTH)/AP
      HAPP=AP*(1.+EP)-REARTH
C CALCULATE VELOCITIES AT END POINTS OF ALL ORBITS.
      VAP0=DSQRT(UK*(1.-E0)/(A0*(1.+E0)))
      VPG0=DSQRT(UK*(1.+E0)/(A0*(1.-E0)))
      VAPT=DSQRT(UK*(1.-ET)/(AT*(1.+ET)))
      VPGT=DSQRT(UK*(1.+ET)/(AT*(1.-ET)))
      VAPP=DSQRT(UK*(1.-EP)/(AP*(1.+EP)))
      VPGP=DSQRT(UK*(1.+EP)/(AP*(1.-EP)))
      IF(IBOUND.EQ.1) GO TO 701
      RI=HPG0+REARTH
      RF=HAPT+REARTH
      GO TO 702
701   RI=HAP0+REARTH
      RF=HPGT+REARTH
702   VI=DSQRT(UK*(2./RI-1./A0))
      VF=DSQRT(UK*(2./RF-1./AT))
      AX=(RI+RF)/2.
      EX=DMAX1(RI,RF)/AX-1.0
      HAPX=AX*(1.+EX)-REARTH
      HPGX=AX*(1.-EX)-REARTH
      VAPX=DSQRT(UK*(2./DMAX1(RI,RF)-1./AX))
      VPGX=DSQRT(UK*(2./DMIN1(RI,RF)-1./AX))
      IF(IBOUND.EQ.1) GO TO 703
      DELTVI=VPGP-VI
      DELTVM=VPGX-VPGP
      DELTVF=VF-VAPX
      GO TO 7035
703    DELTVI=VAPX-VI
      DELTVM=VPGP-VPGX
      DELTVF=VF-VPGP
C FIND FINITE BURN TIMES.
7035  BURN1=-AM0/AMDOT*(DEXP(-AMDOT*DABS(DELTVI)/THRUST)-1.0)
      DELTAM=AMDOT*BURN1
      BURN2=-(AM0-DELTAM)/AMDOT*(DEXP(-AMDOT*DABS(DELTVM)/THRUST)-1.0)
      DELTAM=AMDOT*BURN2+DELTAM
      BURN3=-(AM0-DELTAM)/AMDOT*(DEXP(-AMDOT*DABS(DELTVF)/THRUST)-1.0)
      IF(IBOUND.EQ.1) GO TO 704
      COAST1=TAUP-(BURN1+BURN2)/2.
      COAST2=TAUX/2.-(BURN2+BURN3)/2.
      GO TO 705
704   COAST1=TAUX/2.-(BURN1+BURN2)/2.
      COAST2=TAUP-(BURN2+BURN3)/2.
C SET UP TIMES ARRAY
705   IF(T0-2000..LT.0.0) TT=TT+2000.-T0
      IF(T0-2000.0.LT.0.0) T0=0.0
      IF(T0.GT.0.001) T0=T0-2000.
```

```
      T0=2000.-BURN1/2.+T0
      TBEGIN=T0
      TIMES(1)=T0
      TIMES(2)=T0+BURN1
      TIMES(3)=TIMES(2)+COAST1
      TIMES(4)=TIMES(3)+BURN2
      TIMES(5)=TIMES(4)+COAST2
      TIMES(6)=TIMES(5)+BURN3
C STORE STATES IN X0 AND XT
      DO 710 I=1,3
      X0(I)=R0(I)
      X0(I+3)=V0(I)
      XT(I)=RT(I)
710   XT(I+3)=VT(I)
C SET UP INITIAL Q FOR INBOUND MISSION
      FACTOR=(UK+VAPX*VPGX*(HAPX+REARTH))/((HAPX+REARTH)**3*(VAPX+VPGX))
      DO 720 I=1,3
      Q0(I)=-X0(I+3)/V1
720   Q0(I+3)=+X0(I)*FACTOR
      NO=0
C COAST TUG BACK TO START OF FIRST BURN
      CALL COAST(X0,Q0,-BURN1/2.0,X0,Q0,PHI,PHI)
      IF(IBOUND.EQ.1) GO TO 202
C
C CONVERGE THREE BURN OUTBOUND MISSION BACKWARDS IN TIME
C TO INSURE CONVERGENCE
C
      IBACK=1
C
C SAVE INITIAL STATE OF TUG
      DO 750 I=1,6
750   TUGSAV(I)=X0(I)
      TIMTUG=T0
C PROPAGATE TARGET STATE TO FINAL BURN NODE
      NO=-1
      CALL COAST(XT,QDUM,TAUP+TAUX/2.0,X0,QDUM,PHI,PHI)
C SETUP IMPULSIVE Q SOLUTION AT NODE
      FACTOR=(UK+VAPX*VPGX*(HAPX+REARTH))/((HAPX+REARTH)**3*(VAPX+VPGX))
      DO 760 I=1,3
      Q0(I)=X0(I+3)/VAPT
760   Q0(I+3)=-X0(I)*FACTOR
C PROPAGATE TO END OF LAST BURN
      NO=0
      CALL COAST(X0,Q0,BURN3/2.0,X0,Q0,PHI,PHI)
C SET UP TIMES ARRAY
      TIMES(1)=T0
      TIMES(2)=TIMES(1)+BURN3
      TIMES(3)=TIMES(2)+COAST2
      TIMES(4)=TIMES(3)+BURN2
      TIMES(5)=TIMES(4)+COAST1
      TIMES(6)=TIMES(5)+BURN1
C REDUCE MASS, SET TIMES AND TARGET STATE
      T0=TIMES(1)
      TT=TIMES(6)
      AM0=AM0-(BURN1+BURN2+BURN3)*AMDOT
```

```
C REVERSE MASS RATE,VELOCITY AND UDOT.
      AMDOT=-AMDOT
      VEH(1,2)=-VEH(1,2)
      DO 790 J=4,6
      XT(J-3)=TUGSAV(J-3)
      XT(J)=-TUGSAV(J)
      X0(J)=-X0(J)
790   Q0(J)=-Q0(J)
C
C WRITE IMPULSIVE SOLUTION.
202   WRITE(IOUTPT,3200) NBURNS
3200  FORMAT(//,20X,' * * * * *',I2,'-BURN IMPULSIVE APPROXIMATION SUMMA
     1RY * * * *',///,35X,' * * ORBITAL ELEMENTS * *',//,12X,' SEMI-MAJO
     2R AXIS ECCENTRICITY HEIGHT(APOGEE) HEIGHT(PERIGEE)   PERIOD   V(AP
     3OGEE) V(PERIGEE)',/,14X,'(KILOMETERS)',16X,'(KILOMETERS)     (KILOM
     4ETERS)   (SECONDS)  (KM/SEC)   (KM/SEC)')
      WRITE(IOUTPT,3201) A0,E0,HAP0,HPG0,TAU0,VAP0,VPG0
3201  FORMAT('       TUG',F16.3,F13.5,F16.3,F16.3,F12.2,F9.3,F11.3)
      IF(NBURNS.EQ.3) WRITE(IOUTPT,3202) AP,EP,HAPP,HPGP,TAU ,VAPP,VPGP
3202  FORMAT('  PHASING',F16.3,F13.5,F16.3,F16.3,F12.2,F9.3,F11.3)
      WRITE(IOUTPT,3203) AX,EX,HAPX,HPGX,TAUX,VAPX,VPGX
3203  FORMAT(' TRANSFER',F16.3,F13.5,F16.3,F16.3,F12.2,F9.3,F11.3)
      WRITE(IOUTPT,3204) AT,ET,HAPT,HPGT,TAUT,VAPT,VPGT
3204  FORMAT('    TARGET',F16.3,F13.5,F16.3,F16.3,F12.2,F9.3,F11.3,//)
      WRITE(IOUTPT,3205) RELINC
3205  FORMAT('  RELATIVE INCLINATION',F8.3,' DEGREES (MEASURED + OR - AT
     1 PERIGEE)',///,20X,'* * * * * * * * * * * * * * * * * * * * * *
     2 * *',//)
      WRITE(IOUTPT,3206) TBEGIN,BURN1
3206  FORMAT(' FIRST BURN STARTS AT',F10.2,' SECONDS',/,' FIRST BURN IS'
     1,F12.2,' SECONDS')
      IF(NBURNS.EQ.2) WRITE(IOUTPT,3207) COAST1,BURN2
      IF(NBURNS.EQ.3) WRITE(IOUTPT,3208) COAST1,BURN2,COAST2,BURN3
3207  FORMAT(' COAST IS',F16.2,' SECONDS',/,' SECOND BURN IS',F8.2,' SEC
     1ONDS',//)
3208  FORMAT(' 1ST COAST IS',F13.2,' SECONDS',/,' SECOND BURN IS',F11.2,
     1' SECONDS',/,' SECOND COAST IS',F10.2,' SECONDS',/,' FINAL BURN IS
     2',F12.2,' SECONDS',//)
C CALCULATE THE WEIGHTS,BASED ON ESTIMATED BURN TIME.
204   IF(NOTARG.NE.1) GO TO 205
      READ(INPUT,NAMLS2)
      RELINC=0.0
205   CONTINUE
      TOTBRN=DABS(TIMES(6)+TIMES(4)+TIMES(2)-
     1            (TIMES(5)+TIMES(3)+TIMES(1)))
      PSI=THRUST/(VEH(1,1)-AMDOT*TOTBRN)
      DO 248 I=1,6
      BETA(I)=1.002
      IF(I.GT.3) BETA(I)=1.008
248   WEIGHT(I)=BETA(I)*PSI/(1.+BETA(I)*PSI)
203   CONTINUE
      MODE=1
C CALCULATE END CONDITIONS.
      CALL BVALS(XT,Q0,PTV,TV,-1)
      DO 249 I=1,6
```

```
  249 CC(I)=DD(I)
C * * * * *
C
      IF(ITURNR.EQ.0) WRITE(IOUTPT,3210)
      IF(ITURNR.EQ.1) WRITE(IOUTPT,3213)
      IF(ITURNR.EQ.2) WRITE(IOUTPT,3220)
      IF(ITURNR.EQ.3) WRITE(IOUTPT,3223)
 3223 FORMAT(' BEGIN GUIDANCE-ONLY CONVERGENCE')
 3220 FORMAT(' ADD 3RD BURN AND RECONVERGE.')
 3213 FORMAT(' BEGIN TURN-AROUND CONVERGENCE.')
 3210 FORMAT(' BEGIN COPLANAR CONVERGENCE.',///)
C TRY TO CONVERGE THE COPLANAR MISSION IN .LE. 30 ITERATIONS.
C
      DO 250 ITER=1,30
      NOP=1
      QMAG=DSQRT(Q0(1)**2+Q0(2)**2+Q0(3)**2)
      DO 247 I=1,6
  247 Q0(I)=Q0(I)/QMAG
      CALL GUIDE(0.0)
      CALL AUXOUT
      CALL CKSET(CK)
      DQ0MAX=0.0
      DTMAX=0.0
      DO 251 I=1,6
      Q0(I)=Q0(I)+DQ0(I)*CK
      TIMES(I)=TIMES(I)+DTIMES(I)*CK
      IF(DABS(DQ0(I)).GT.DQ0MAX) DQ0MAX=DABS(DQ0(I))
  251 IF(DABS(DTIMES(I)).GT.DTMAX) DTMAX=DABS(DTIMES(I))
      AM1=AM0
      IF(IBACK.EQ.1)AM0=VEH(1,1)+(TIMES(6)-TIMES(5)+TIMES(4)_TIMES(3)
     1 +TIMES(2)-TIMES(1))*AMDOT
      AM0=(AM0+AM1)*.5
      IF(DTMAX.LT.1.D-6*DABS(TIMES(6)).AND.DQ0MAX.LT..001.AND.
     1DABS(AM0-AM1).LT.VEH(1,1)*1.D-6) GO TO 252
 3211 FORMAT(' COPLANAR MISSION CONVERGED IN', I3,' ITERATIONS.')
 3214 FORMAT(' TURN-AROUND ACHIEVED IN',I3,' ITERATIONS.')
 3221 FORMAT(' 3RD BURN ADDED AND CONVERGED IN',I3,' ITERATIONS.')
  250 CONTINUE
C DID NOT CONVERGE IN 30 ITERATIONS. DUMP VARIABLES AND STOP.
      WRITE(IOUTPT,3005)
 3005 FORMAT(' PLANAR MISSION DID NOT CONVERGE IN 30 ITERATIONS. STOP')
      STOP
  252 CONTINUE
      IF(ITURNR.EQ.0) WRITE(IOUTPT,3211) ITER
      IF(ITURNR.EQ.1) WRITE(IOUTPT,3214) ITER
      IF(ITURNR.EQ.2) WRITE(IOUTPT,3221) ITER
      IF(ITURNR.EQ.3) WRITE(IOUTPT,3222) ITER
 3222 FORMAT(' GUIDANCE-ONLY CONVERGENCE ACHIEVED IN',I3,' ITERATIONS.')
C
C ADD PLANE CHANGE
C
      ISECD=0
      ANGLE=0.0
C IF NO PLANE CHANGE SKIP AROUND
      IF(DABS(RELINC).LT.RERROR) GO TO 295
```

```
C TRANSFORM TO 2-BURN MISSION FOR INSERTION OF PLANE CHANGE
      IF(NBURNS.NE.3) GO TO 2540
      T6SAV=TIMES(6)
      T5SAV=TIMES(5)
      TIMSAV=TT
      DO 253 I=1,4
  253 TIMES(7-I)=TIMES(5-I)
      TIMES(2)=0.0
      TIMES(1)=0.0
      NOP=0
C CALL GUIDE TO GENERATE PHASING ORBIT END CONDITIONS
      CALL GUIDE(0.0)
      DO 254 I=1,6
  254 XPHASE(I)=X(I)
      TT=TIMES(6)
 2540 CONTINUE
C PERFORM REQUIRED PLANE CHANGE IN 10 DEGREE STEPS
      RELT=DABS(RELINC)
  255 ANGL=DMIN1(DABS(ANGLE)+10.,RELT)
      ANGL=DSIGN(ANGL,RELINC)
      DANGL=ANGL-ANGLE
      ANGLE=ANGL
      WRITE(IOUTPT,4000) ANGLE
 4000 FORMAT('  ATTEMPT TO CONVERGE ',F6.2,' DEGREE PLANE CH NGE')
C TEST WHETHER TARGET STATE AT ONE OF NODES(ONLY IF PHASE NOT CALLED)
      IF(IPHASE.NE.0) GO TO 257
C RT,VT AT APOGEE OR PERIGEE,ROTATE THROUGH ANGLE
      DO 256 I=1,3
  256 XT(I)=RT(I)
      XT(4)=0.0
      XT(5)=VT(2)*DCOS(ANGLE*DEGCON)
      XT(6)=VT(2)*DSIN(ANGLE*DEGCON)
      GO TO 1000
C CHECK TO SEE IF CONVERGING BACKWARDS
  257 IF(IBACK.EQ.0) GO TO 2570
C CONVERGING BACKWARDS ROTATE X0 AND Q0
      CALL ROTATE(X0,PRGEE,DANGL*DEGCON,6)
      CALL ROTATE(Q0,PRGEE,DANGL*DEGCON,6)
      GO TO 259
C CONVERGING FORWARDS,FIND IF 2 OR 3 BURNS
 2570 IF(NBURNS.EQ.3) GO TO 258
C 2-BURN MISSION,ROTATE PRESENT XT
      CALL ROTATE(XT,PRGEE,DANGL*DEGCON,6)
      GO TO 1000
C 3-BURN FORWARD ROTATE XPHASE
  258 CALL ROTATE(XPHASE,PRGEE,DANGL*DEGCON,6)
C SET UP 3-BURN AS 2-BURN
  259 DO 2600 I=1,6
 2600 XT(I)=XPHASE(I)
C ON FIRST PASS SET UP AN INITIAL COAST
      IF(ISECD.EQ.1) GO TO 1000
      ISECD=1
      NO=0
      TBCT=-500.
      CALL COAST(X0,Q0,TBCT,X0,Q0,PHI,PHI)
```

```
      T0=T0+TbCT                                                        MAI00826
C REDUCE INITIAL MASS TO REFLECT ESTIMATED THIRD BURN                   MAI00827
      IF(IBACK.EQ.0) GO TO 1000                                         MAI00828
      AMSAV=AMDOT*(T6SAV-T5SAV)                                         MAI00829
      VEH(1,1)=VEH(1,1)+AMSAV                                           MAI00830
 1000 CONTINUE                                                          MAI00831
C SET UP END CONDITIONS FOR PLANE CHANGE                                MAI00832
      CALL BVALS(XT,Q0,PTV,TV,-1)                                       MAI00833
      DO 261 I=1,6                                                      MAI00834
  261 CC(I)=DD(I)                                                       MAI00835
C * * * * *                                                             MAI00836
C                                                                       MAI00837
C                                                                       MAI00838
C TRY TO CONVERGE WITH PLANE CHANGE IN .LE. 30 ITERATIONS.              MAI00839
      DO 290 ITER=1,30                                                  MAI00840
      NOP=1                                                             MAI00841
      QMAG=DSQRT(Q0(1)**2+Q0(2)**2+Q0(3)**2)                            MAI00842
      DO 289 I=1,6                                                      MAI00843
  289 Q0(I)=Q0(I)/QMAG                                                  MAI00844
      CALL GUIDE(0.0)                                                   MAI00845
      CALL AUXOUT                                                       MAI00846
      CALL CKSET(CK)                                                    MAI00847
      DQ0MAX=0.0                                                        MAI00848
      DTMAX=0.0                                                         MAI00849
      IPRINT=0                                                          MAI00850
      DO 288 I=1,6                                                      MAI00851
      Q0(I)=Q0(I)+DQ0(I)*CK                                             MAI00852
      TIMES(I)=TIMES(I)+DTIMES(I)*CK                                    MAI00853
      IF(DABS(DQ0(I)).GT.DQ0MAX) DQ0MAX=DQ0(I)                          MAI00854
  288 IF(DABS(DTIMES(I)).GT.DTMAX) DTMAX=DABS(DTIMES(I))                MAI00855
      AM1=AM0                                                           MAI00856
      IF(IBACK.EQ.1)AM0=VEH(1,1)+(TIMES(6)-TIMES(5)+TIMES(4) -TIMES(3)  MAI00857
     1 +TIMES(2)-TIMES(1))*AMDOT                                        MAI00858
      AM0=(AM1+AM0)*.5                                                  MAI00859
      IF(DTMAX.LT.1.D-6*DABS(TIMES(6)).AND.DQ0MAX.LT..001.AND.          MAI00860
     1 DABS(AM1-AM0).LT.VEH(1,1)*1.D-6) GO TO 291                       MAI00861
  290 CONTINUE                                                          MAI00862
C DID NOT CONVERGE IN 30 ITERATIONS. DUMP VARIABLES AND STOP.           MAI00863
      WRITE(IOUTPT,3006)                                                MAI00864
 3006 FORMAT(' OUT-OF-PLANE MISSION DID NOT CONVERGE IN 30 ITERATIONS. SMAI00865
     1TOP.')                                                            MAI00866
      STOP                                                              MAI00867
  291 CONTINUE                                                          MAI00868
      WRITE(IOUTPT,3212) ANGLE,ITER                                     MAI00869
 3212 FORMAT(F7.2,' DEGREE PLANE CHANGE CONVERGED IN',I3,' ITERATIONS.')MAI00870
      IF(DABS(ANGLE).LT.DABS(RELINC)) GO TO 255                         MAI00871
C CHANGE BACK TO 3-BURN IF NECESSARY                                    MAI00872
      IF(NBURNS.NE.3) GO TO 200                                         MAI00873
      DO 292 I=1,4                                                      MAI00874
  292 TIMES(I)=TIMES(I+2)                                               MAI00875
      TIMES(5)=T5SAV                                                    MAI00876
      TIMES(6)=T6SAV                                                    MAI00877
      DO 293 I=1,3                                                      MAI00878
      XT(I+3)=-TUGSAV(I+3)                                              MAI00879
  293 XT(I)=TUGSAV(I)                                                   MAI00880
```

```
      TT=TIMSAV
      RELINC=0.0
C ADD MASS TO INITIAL MASS IF BACKWARDS MISSION
      IF(IBACK.EQ.1)VEH(1,1)=VEH(1,1)-AMSAV
      ITURNR=2
C ROTATE XT IF FORWARDS MISSION
      IF(IBACK.EQ.1) GO TO 203
      DO 294 I=1,3
      XT(I)=RT(I)
294   XT(I+3)=VT(I)
      CALL ROTATE(XT,PRGEL,ANGLE*DEGCON,6)
      GO TO 203
C TEST TO SEE IF MISSION TURNAROUND IS NECESSARY
295   CONTINUE
      IF(IBACK.NE.1) GO TO 260
C
C TURNAROUND MISSION AND RECONVERGE
C
C SET UP TIMES ARRAY,Q0,AM0 AND XT
      WRITE(IOUTPT,4001)
4001  FORMAT('  TURNAROUND MISSION')
      B1=TIMES(6)-TIMES(5)
      C1=TIMES(5)-TIMES(4)
      B2=TIMES(4)-TIMES(3)
      C2=TIMES(3)-TIMES(2)
      B3=TIMES(2)-TIMES(1)
      TALIGN=TT-TIMES(6)
      TTLGN=TIMES(6)-T0
      T0=TALIGN+TIMTUG
      TIMES(1)=T0
      TIMES(2)=TIMES(1)+B1
      TIMES(3)=TIMES(2)+C1
      TIMES(4)=TIMES(3)+B2
      TIMES(5)=TIMES(4)+C2
      TIMES(6)=TIMES(5)+B3
      TT=T0+TTLGN
      AMDOT=-AMDOT
      VEH(1,2)=-VEH(1,2)
      DO 310 I=1,3
      Q0(I)=Q(I)
      Q0(I+3)=-Q(I+3)
      XT(I)=X0(I)
310   XT(I+3)=-X0(I+3)
      AM0=VEH(1,1)
C COAST TUGSAV TO ALIGN WITH TIMES(1)
      NO=-1
      CALL COAST(TUGSAV,QDUM,TALIGN,X0,QDUM,PHI,PHI)
C RECONVERGE WITH FORWARD MISSION
      IBACK=0
      RELINC=0.0
      ITURNR=1
      GO TO 203
260   CONTINUE
C VERIFY FINAL ORBIT.
      DO 262 I=1,3
```

```
      RTA(I)=X(I)                                                       MAI00936
262   VTA(I)=X(I+3)                                                     MAI00937
      CALL ELMNTS(RTA,VTA,AA,EA,HA,PGA,TAUA)                            MAI00938
      WRITE(IOUTPT,3215) RTA,VTA,AA,EA,HA,PGA,TAUA                      MAI00939
3215  FORMAT(' ORBIT ACTUALLY ACHIEVED:',/,                             MAI00940
     1' POSITION=',3D14.6,/,' VELOCITY=',3D14.6,/,                      MAI00941
     2' SEMI-MAJOR AXIS=',F10.2,/,' ECCENTRICITY=',F8.6,/,              MAI00942
     3' H-VECTOR=',3D14.6,/,' PERIGEE=',3D14.6,/,' PERIOD=',.10.2)      MAI00943
      BUR1=TIMES(2)-TIMES(1)                                            MAI00944
      BUR2=TIMES(4)-TIMES(3)                                            MAI00945
      BUR3=TIMES(6)-TIMES(5)                                            MAI00946
      COAS0=TIMES(3)-T0                                                 MAI00947
      COAS1=TIMES(1)-T0                                                 MAI00948
      COAS2=TIMES(3)-TIMES(2)                                           MAI00949
      COAS3=TIMES(5)-TIMES(4)                                           MAI00950
      IF(NBURNS.EQ.2) WRITE(IOUTPT,3217)COAS0,BUR2,COAS3,BUR3           MAI00951
3217  FORMAT(//,' CONVERGED COASTS AND BURNS FOR 2-BURN MISSION:',/,    MAI00952
     1' INITIAL COAST=',F10.2,/,'   FIRST BURN=',F10.2,/,'  SECOND COASTMAI00953
     2=',F10.2,/,'    FINAL BURN=',F10.2)                               MAI00954
C                                                                       MAI00955
      IF(NBURNS.EQ.3) WRITE(IOUTPT,3218) COAS1,BUR1,COAS2,BUR2,COAS3,BURMAI00956
     13                                                                 MAI00957
3218  FORMAT(' CONVERGED COASTS AND BURNS FOR 3-BURN MISSION:',/,       MAI00958
     1' INITIAL COAST=',F10.2,/,'   FIRST BURN=',F10.2,/,'  SECOND COASTMAI00959
     2=',F10.2,/,'   SECOND BURN=',F10.2,/,'   THIRD COAST=',F10.2,/,   MAI00960
     3'   FINAL BURN=',F10.2)                                           MAI00961
C                                                                       MAI00962
C GUIDANCE SECTION                                                      MAI00963
C                                                                       MAI00964
C                                                                       MAI00965
C CHECK IF GUIDANCE DESIRED.                                            MAI00966
      IF(NOTARG.EQ.-1) STOP                                             MAI00967
C CHECK IF 2 OR 3 BURNS.                                                MAI00968
      IF(NBURNS.EQ.2) GO TO 410                                         MAI00969
C 3 BURN MISSION. REMOVE ANY INITIAL COAST.                             MAI00970
      IF(DABS(T0-TIMES(1)).LT.TERROR) GO TO 405                         MAI00971
      N0=0                                                              MAI00972
      CALL COAST(X0,Q0,TIMES(1)-T0,X0,Q0,PHI,PHI)                       MAI00973
      T0=TIMES(1)                                                       MAI00974
405   AM0=VEH(1,1)                                                      MAI00975
      T0INT=TIMES(1)                                                    MAI00976
      IF(IBOUND.EQ.1) GO TO 410                                         MAI00977
C TURN AROUND OUTBOUND 3-BURN MISSION.                                  MAI00978
C                                                                       MAI00979
C SET UP STATE AND TIME ARRAYS.                                         MAI00980
      DO 406 I=1,3                                                      MAI00981
      XTS(I)=X0(I)                                                      MAI00982
      XTS(I+3)=X0(I+3)                                                  MAI00983
      QTS(I)=Q0(I)                                                      MAI00984
      QTS(I+3)=-Q0(I+3)                                                 MAI00985
      X0(I+3)=-X(I+3)                                                   MAI00986
      Q0(I+3)=-Q(I+3)                                                   MAI00987
      XT(I+3)=-XTS(I+3)                                                 MAI00988
      X0(I)=X(I)                                                        MAI00989
      Q0(I)=Q(I)                                                        MAI00990
```

```
 406  XT(I)=XTS(I)
      B1=TIMES(2)-TIMES(1)
      C1=TIMES(3)-TIMES(2)
      B2=TIMES(4)-TIMES(3)
      C2=TIMES(5)-TIMES(4)
      B3=TIMES(6)-TIMES(5)
      TIMES(1)=2000.
      TIMES(2)=TIMES(1)+B3
      TIMES(3)=TIMES(2)+C2
      TIMES(4)=TIMES(3)+B2
      TIMES(5)=TIMES(4)+C1
      TIMES(6)=TIMES(5)+B1
      T0=TIMES(1)
      TT=TIMES(6)
      BURNT=DABS(TIMES(6)-TIMES(5)+TIMES(4)-TIMES(3)+
     1TIMES(2)-TIMES(1))
      AM0=VEH(1,1)-BURNT*VEH(1,2)
      AMDOT=-AMDOT
      VEH(1,2)=-VEH(1,2)
C SET UP INITIAL COAST FOR BACKWARDS MISSION.
      NO=0
      CALL COAST(X0,Q0,-500.,X0,Q0,PHI,PHI)
      T0=T0-500.
C SET UP MONTE CARLO RUNS.
C
C
C CALCULATE END CONDITIONS
 410  CALL BVALS(XT,QDUM,PTV,TV,-1)
      DO 411 I=1,6
 411  CC(I)=DD(I)
C SAVE INITIAL CONDITIONS FOR NEXT MONTE CARLO RUN.
      DO 415 I=1,6
      X0S(I)=X0(I)
      Q0S(I)=Q0(I)
      VEHS(I)=VEH(1,I)
      TIMESS(I)=TIMES(I)
      XTS(I)=XT(I)
 415  CCS(I)=CC(I)
      VEHS(7)=VEH(1,7)
      T0S=T0
      TTS=TT
      AM0S=AM0
      IBOUNS=IBOUND
C LOOP FOR MONTE CARLO RUNS.
      IPRINT=1
      DO 420 MONTE=1,MCARLO
      IUPDAT=0
C RESTORE VARIABLES.
      DO 425 I=1,6
      VEH(1,I)=VEHS(I)
      X0(I)=X0S(I)
      Q0(I)=Q0S(I)
      XT(I)=XTS(I)
      QT(I)=QTS(I)
      TIMES(I)=TIMESS(I)
```

```
  425 CC(I)=CCS(I)                                              MAI01046
      VEH(1,7)=VEHS(7)                                          MAI01047
      DO 426 I=2,10                                             MAI01048
      DO 426 J=1,7                                              MAI01049
  426 VEH(I,J)=0.0                                              MAI01050
      IBOUND=IBOUNS                                             MAI01051
      TRUEMS=VEH(1,1)                                           MAI01052
      TCLOCK=0.0                                                MAI01053
      TACCUM=0.0                                                MAI01054
      AMO=AMOS                                                  MAI01055
      TO=TOS                                                    MAI01056
      TT=TTS                                                    MAI01057
      IF(NBURNS.EQ.3) CALL BCBCB(IBOUND,TOINT)                  MAI01058
  420 IF(NBURNS.EQ.2) CALL CBCB                                 MAI01059
      CALL STATIS(MCARLO)                                       MAI01060
      STOP                                                      MAI01061
      END                                                       MAI01062
```

Subroutine AUXOUT

A. Purpose

AUXOUT prints the status of the convergence, from the most recent call to GUIDE.

B. Input/Output Definition

Input Parameter	Symbol	Definition
X(I) I=1,6	$\bar{x}$	Vehicle final state
XTF(I) I=1,6	$\bar{x}_T$	Target state at same time as above
TIMES(I) I=1,6	-	Array of times at ends of coast and burn arcs
QØ(I) I=1,6	$\bar{q}_0$	Costate at start of mission
DTIMES(I) I=1,6	$\Delta\bar{t}$	Requested corrections to TIMES
DQØ(I) I=1,6	$\Delta\bar{q}_0$	Requested corrections to costate QØ
IOUTPT	-	Output device number

Output Parameter

None.

C. Method of Computation

The only variable calculated is the estimate of the total burn remaining

$$\text{COST} = |(\text{TIMES}(2) - \text{TIMES}(1)) + (\text{TIMES}(4) - \text{TIMES}(3)) + (\text{TIMES}(6) - \text{TIMES}(5))|$$

FILE: AUXOUT FORTRAN P1 CAMBRIDGE MONITOR SYSTEM

```
      SUBROUTINE AUXOUT
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /GIDIN/XT(6),TT,X0(6),T0,AM0,VEH(10,7),Q0(6),TIMES(6),C(6)
      COMMON /ODEVIC/IOUTPT
      COMMON /GIDOUT/DQ0(6),DTIMES(6),L(12,12),DC(12),X(6),Y(6),
     1Z(12,12),D(6),DUMM(4),SM
      COMMON /BVLOUT/XTF(6),DELTC(6)
      WRITE(IOUTPT,1)X,XTF
    1 FORMAT(//,' X(OBTAINED)=',6E14.6,/,'  X(DESIRED)=',6E14.6)
      COST=DABS(TIMES(6)-TIMES(5)+TIMES(4)-TIMES(3)+TIMES(2)-TIMES(1))
      WRITE(IOUTPT,2) COST
    2 FORMAT(' REMAINING BURN=',D14.6)
      WRITE(IOUTPT,3) Q0,DQ0,TIMES,DTIMES
    3 FORMAT(1X,' Q0=',6E16.8,/,' DQ0=',6E16.8,/,5X,'T=',
     1 6E16.8,/,4X,'DT=',6E16.8,//)
      RETURN
      END
```

Subroutine BCBCB

A. Purpose

Subroutine BCBCB is used during guidance mode to take the vehicle through the first burn of a 3-burn mission. It operates in either a backwards mode (outbound mission) or a normal mode, and is called by MAIN at the start of each Monte-Carlo run. It in turn calls FORWRD at regular intervals until the end of the first burn, at which time it changes mode (if backwards) to the normal mode and calls CBCB to handle the remaining coasts and burns. BCBCB also modifies the TIMES array on each cycle to reflect the fact that part of the first burn has occurred, calls GUIDE to reconverge the mission with the new (possibly perturbed) vehicle state, and adds the resulting corrections to the TIMES array and costate. On the indicated cycles (IOUT = 1 or next-to-last cycle in the burn arc), subroutine NAVOUT is called to collect the Monte-Carlo statistics. On the last cycle in the burn arc, the call to GUIDE (and the addition of the corrections to TIMES and QØ) is skipped and CBCB is called with an initial step time of zero.

B. Input/Output Definition

Input Parameter	Symbol	Definition
IBOUND	-	0 - outbound mission (implies backwards mode) 1 - inbound mission
TØINT	-	In backwards mode, the actual value of TØ
TRUEMS	-	Vehicle mass before start of burn (normally equivalent to AMØ except when in backwards mode)
XT(I) I=1,6	$\bar{x}_T$	Vehicle state in backwards mode
TT	t_T	Time at start of first burn in outbound case
TØ	t_0	Time at start of first burn in inbound case
IOUTPT	-	Output device number

Input Parameter	Symbol	Definition
DTYPE(I) I=1,2	-	I=1; normal guidance step size during burn I=2; not used in BCBCB
TIMES(I) I=1,6	-	Vector of times at end of each leg (or start of each leg in backwards mode)

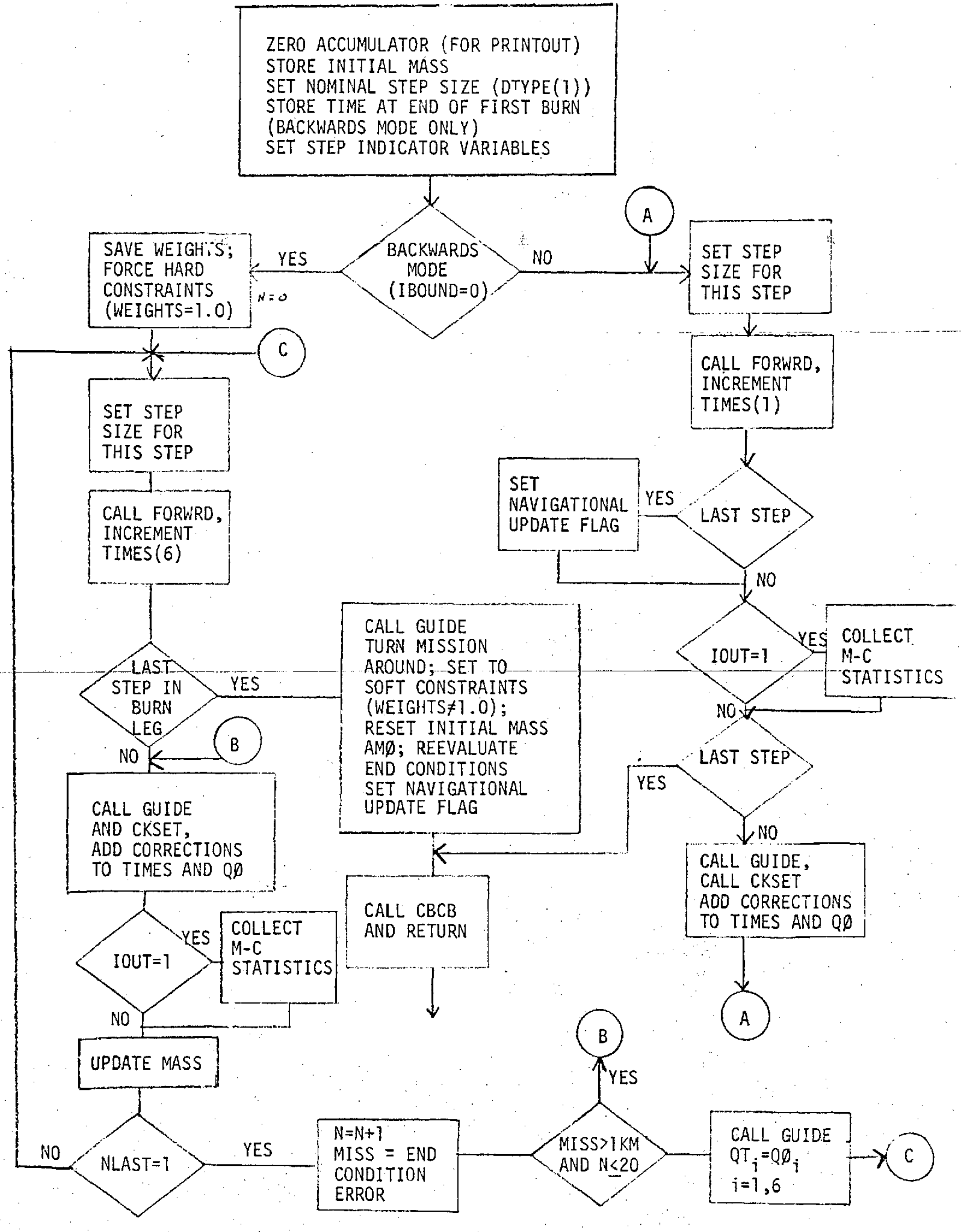
Output Parameter	Symbol	Definition
IPRINT	-	Always 0; shuts off printout resulting from calls to GUIDE after first step in first Monte-Carlo run
MODE	-	Always 0; restores mission to free-time rendezvous (backwards mode only)
AMØ	-	Mass at end of first burn
TIMES(I) I=1,6	-	Vector of times at end of arcs, with first burn deleted from the vector (TIMES(1)=0, TIMES(2)=0) and, in backwards mode, the vector restored to its normal form
QØ(I) I=1,6	$\bar{q}_0$	Costate at end of first burn
TØ	-	Time at end of first burn
TT	-	Time for which target state is valid
CC(I) I=1,6	-	New end conditions for target (backwards mode only)

C. Method of Computation

After zeroing the time accumulator (used to determine when Monte-Carlo statistics are to be collected), saving the vehicle initial mass, and initializing several control integers, BCBCB branches to one of two separate sections of code, depending on whether a normal 3-burn mission is being run. In either case, it is assumed that the first burn begins immediately, with no initial coast.

In the backwards mode, the TIMES array as supplied to BCBCB is already reversed and ready to use, as are TØ and TT. The weights are set to 1.0 since the backwards mode works best with hard constraints and mode is set to 3 to change to a fixed time rendezvous. Subroutine FORWRD is then called every DTYPE(1) seconds during the first burn, with the exception of the last two steps which are approximately equal to each other and less than DTYPE(1)/2, and TIMES(6) is updated. Each time the print accumulator exceeds PTB, subroutine NAVOUT is called to collect Monte Carlo statistics, and the accumulator is reset to zero. M/C statistics are also collected on the next-to-last step in the burn arc. Also, following each call to FORWRD, except the last, GUIDE is called and the corrections are added to QØ and TIMES, QØ is maintained at unit magnitude, and the estimate of vehicle final mass is recalculated from the mass rate, current mass, and requested changes in the burn times. On the next-to-last call to FORWRD (NLAST=1), subroutine GUIDE is called repeatedly (with no changes in vehicle state) until the miss in final position is less than 1 kilometer. On the last call to FORWRD, GUIDE is called but no changes are permitted in the TIMES array and QØ and the weights are restored to their original value. In addition, the flag is set to add the navigation update corrections to vehicle state on the very first call to FORWRD from CBCB. The mission is then turned around to normal mode, and the target end conditions reevaluated. Finally, subroutine CBCB is called to handle the remaining coasts and burns.

In normal mode, BCBCB works in much the same way, except that the states and TIMES array are not reversed, and TØ is updated rather than TIMES(6).


ZERO ACCUMULATOR (FOR PRINTOUT)
STORE INITIAL MASS
SET NOMINAL STEP SIZE (DTYPE(1))
STORE TIME AT END OF FIRST BURN
(BACKWARDS MODE ONLY)
SET STEP INDICATOR VARIABLES
BACKWARDS MODE (IBOUND=0)
YES
NO
N=0
SAVE WEIGHTS; FORCE HARD CONSTRAINTS (WEIGHTS=1.0)
A
SET STEP SIZE FOR THIS STEP
C
CALL FORWRD, INCREMENT TIMES(1)
SET STEP SIZE FOR THIS STEP
SET NAVIGATIONAL UPDATE FLAG
LAST STEP
CALL FORWRD, INCREMENT TIMES(6)
CALL GUIDE
TURN MISSION AROUND; SET TO SOFT CONSTRAINTS (WEIGHTS≠1.0);
RESET INITIAL MASS AMØ; REEVALUATE END CONDITIONS
SET NAVIGATIONAL UPDATE FLAG
IOUT=1
COLLECT M-C STATISTICS
LAST STEP IN BURN LEG
B
LAST STEP
CALL GUIDE AND CKSET, ADD CORRECTIONS TO TIMES AND QØ
CALL GUIDE, CALL CKSET ADD CORRECTIONS TO TIMES AND QØ
CALL CBCB AND RETURN
IOUT=1
COLLECT M-C STATISTICS
A
B
UPDATE MASS
NLAST=1
N=N+1
MISS = END CONDITION ERROR
MISS>1KM AND N≤20
CALL GUIDE
QT_i=QØ_i
i=1,6
C

```
      SUBROUTINE BCBCB(IBOUND,IOINT)
C SUBROUTINE TO TAKE THE TUG THROUGH THE INITIAL BURN OF A 3 BURN
C MISSION IN GUIDANCE MODE. WORKS FOR BOTH INBOUND AND OUTBOUND MISSIONS
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BVLOUT/XF(6)
      COMMON /UPDATE/IUPDAT
      COMMON /CPHYS/UK,REARTH,DUM1,DUM2,DUM3
      COMMON /ONLINE/IPRINT
      COMMON /CMODE/MODE
      COMMON /CNAV/TRUEMS,FNAV,TACCUM,IOUT
      COMMON /GIDIN/XT(6),TT,X0(6),T0,AM0,VEH(10,7),Q0(6),TIMES(6),CC(6)
      COMMON /GIDOUT/DQ0(6),DTIMES(5),E(12,12),DC(12),X(6),Q(6),Z(12,12)
     1,DD(6),SM(5)
      COMMON /CINDEX/NARC,IARC,JMAX,JM,JMAX1,JLAST,NO,NOP,NRIGOS
      COMMON /CPERT/PERT(3),DTYPE(2),DFAC
      COMMON /CJ/BETA(6),PSI,UF,CK,TBOUND,UBOUND
      COMMON /CWT/WEIGHT(6)
      COMMON /CQCST/QI(6)
      COMMON /XQSAVE/XTS(6),QTS(6)
      DIMENSION WTS(6)
C ZERO TIME ACCUMULATOR.
      TACCUM=0.0
      IOUT=0
C SAVE INITIAL MASS
      TOTLMS=TRUEMS
C SET NOMINAL STEP SIZE IN BURN
      DT=DTYPE(1)
C SAVE TIME AT END OF LAST BURN (=TIME AT START OF 1ST BURN FORWARDS)
      IF(IBOUND.EQ.0) T6SAVE=TIMES(6)
C SET VARIABLES
      NLAST=0
      LAST=0
      NPOINT=0
      IF(IBOUND.EQ.1) GO TO 100
C
C
C 3-BURN OUTBOUND MISSION. NOW IN BACKWARDS MODE.
C
C
C SET WEIGHTS.
C WEIGHT(I)=1 REFLECTS HARD CONSTRAINTS ON BACKWARDS BURN.
      DO 1 I=1,6
      WTS(I)=WEIGHT(I)
    1 WEIGHT(I)=1.0
      NC=0
C START MAIN GUIDANCE LOOP FOR FIRST BURN.
    3 IF(NLAST.EQ.1) LAST=1
      IF(DABS(TIMES(6)-TIMES(5)).LE.2.*DTYPE(1))NLAST=1
      IF(NLAST.EQ.1) IOUT=1
      IF(NLAST.EQ.1) DT=DABS(TIMES(6)-TIMES(5))/2.
      IF(LAST.EQ.1) DT=DABS(TIMES(6)-TIMES(5))
      CALL FORWRD(0,-DT,0)
      TIMES(6)=TIMES(6)-DT
      IF(LAST.EQ.1) GO TO 6
   33 NOP=1
```

```
      MODE=3
      CALL GUIDE(0.0)
      IPRINT=0
      CK=-1.0
      CALL CKSET(CK)
C SET INDEX FOR OUTPUT.
      IF(IOUT.EQ.1) NPOINT=NPOINT+1
      IF(IOUT.EQ.1) CALL NAVOUT(1,NPOINT)
C APPLY CORRECTIONS TO Q0. KEEP QT (=TRUE Q0) AT UNIT MAGNITIDE.
C QT IS THE ACTUAL COSTATE WHICH WOULD BE USED FOR STEERING.
      DO 4 I=1,6
      Q0(I)=Q0(I)+CK*DQ0(I)
4     QT(I)=Q(I)
      UMAG=DSQRT(Q0(1)**2+Q0(2)**2+Q0(3)**2)
      DO 41 I=1,6
41    Q0(I)=Q0(I)/UMAG
C APPLY CORRECTIONS TO ALL TIMES EXCEPT THE LAST, SINCE IT IS REALLY
C THE CLOCK TIME IN BACKWARDS MODE.
      DO 5 I=1,5
5     TIMES(I)=TIMES(I)+CK*DTIMES(I)
      AM1=TRUEMS+VEH(1,2)*(TIMES(6)-TIMES(5)+TIMES(4)-TIMES(3)+TIMES(2)-
     1TIMES(1))
      AM0=(AM0+AM1)*.5
      IF(NLAST.NE.1) GO TO 3
      NC=NC+1
      IF(NC.EQ.1) GO TO 33
      DMISS=DSQRT((X(1)-XF(1))**2+(X(2)-XF(2))**2+(X(3)-XF(3))**2)
      IF(DMISS.GT.1..AND.NC.LE.20) GO TO 33
      CALL GUIDE(0.0)
      DO 11 I=1,6
11    QT(I)=Q(I)
      GO TO 3
C END OF FIRST BURN. TURN MISSION AROUND.
6     CALL GUIDE(0.0)
      MODE=0
C SET FLAG TO SIGNAL NAVIGATIONAL UPDATE.
      IUPDAT=1
C RESTORE THE WEIGHTS.
      DO 7 I=1,6
7     WEIGHT(I)=WTS(I)
C CHANGE OUTBOUND 3-BURN MISSION BACK TO NORMAL MODE.
      B1=DABS(TOSAVE-TIMES(5))
      B2=DABS(TIMES(4)-TIMES(3))
      B3=DABS(TIMES(2)-TIMES(1))
      C2=DABS(TIMES(5)-TIMES(4))
      C3=DABS(TIMES(3)-TIMES(2))
      TT=TIMES(1)-T0
      T0=B1+T0INT
      TIMES(1)=0.0
      TIMES(2)=0.0
      TIMES(3)=T0+C2
      TIMES(4)=TIMES(3)+B2
      TIMES(5)=TIMES(4)+C3
      TIMES(6)=TIMES(5)+B3
      TT=TIMES(6)+TT
```

```
      VEH(1,2)=-VEH(1,2)
      AMO=TOTLMS-B1*VEH(1,2)
      DO 8 I=1,3
      Q0(I)=QT(I)
      Q0(I+3)=-QT(I+3)
      XTEMP1=XT(I)
      XTEMP2=XT(I+3)
      XT(I)=X0(I)
      XT(I+3)=-X0(I+3)
      X0(I)=XTEMP1
    8 X0(I+3)=-XTEMP2
C SET END CONDITIONS.
      CALL BVALS(XT,Q0,PIV,TV,-1)
      DO 9 I=1,6
    9 CC(I)=DD(I)
      CALL CBCB
      RETURN
C
C
C 3-BURN INBOUND MISSION (FORWARD MODE).
C
C
C SET Q0 TO UNIT MAGNITUDE.
  100 QMAG=DSQRT(Q0(1)**2+Q0(2)**2+Q0(3)**2)
      DO 101 I=1,6
  101 Q0(I)=Q0(I)/QMAG
  102 IF(NLAST.EQ.1) LAST=1
      IF(TIMES(2)-TIMES(1).LE.2.*DTYPE(1)) NLAST=1
      IF(NLAST.EQ.1) DT=(TIMES(2)-TIMES(1))/2.
      IF(LAST.EQ.1) DT=TIMES(2)-TIMES(1)
      CALL FORWRD(0,DT,1)
      IF(LAST.EQ.1) IUPDAT=1
      TIMES(1)=TIMES(1)+DT
      IF(LAST.EQ.1) GO TO 106
      NOP=1
      MODE=0
      CALL GUIDE(0,0)
      IPRINT=0
      CK=-1.0
      CALL CKSET(CK)
C ADD CORRECTIONS TO Q0, TIMES.
      DO 103 I=1,6
      Q0(I)=Q0(I)+CK*DQ0(I)
  103 TIMES(I)=TIMES(I)+DTIMES(I)*CK
      IF(NLAST.EQ.1.OR.IOUT.EQ.1) NPOINT=NPOINT+1
      IF(NLAST.EQ.1.OR.IOUT.EQ.1) CALL NAVOUT(1,NPOINT)
      GO TO 100
  106 CALL CBCB
      RETURN
      END
```

Subroutine BVAL5

A. Purpose

The new BVAL5 subroutine replaces the BVAL5 and BVAL6 subroutines in GUIDE 71/6. It calculates the miss in end conditions and partial derivatives of the end conditions for either hard or soft constraint missions with up to six end condition constraints and free or fixed terminal time. The subroutine can also be called (for example, for initializing desired h and e) with NBVAL=-1 to calculate the three components of the angular momentum vector h and the three components of the eccentricity vector e, pointing toward perigee with magnitude of eccentricity. BVAL5 calls the subroutine COAST to obtain target state XTF at the end of the mission, TIMES(6).

B. Input/Output Definition

Input Parameter	Symbol	Definition
XF(I) for I=1 to 3	r	Final vehicle position
for I=4 to 6	v	Final vehicle velocity
QF(I) for I=1 to 3	u	Final control vector
for I=4 to 6	$\dot{u}$	Final (du/dt)
PTV(I) for I=1 to 12	$\left(\frac{\partial T_V}{\partial y}\right)$	Partial derivatives of T_V with respect to $y=(r^T,v^T,u^T,\dot{u}^T)^T$ evaluated in BUZZ
TV	T_V	Phasing transversality condition $\mu(r^Tu)/\lvert r\rvert^3 + (v^T\dot{u})$ evaluated in BUZZ
NBVAL	-	Flag parameter indicating whether or not miss in end conditions and their derivatives are to be calculated
UK	μ	Gravitational constant
C(I) for I=1 to 3	h_d	Desired orbital angular velocity
for I=4 to 6	e_d	Desired eccentricity vector

Input Parameter	Symbol	Definition
Z(I,J) I=1 to 12, J=1 to JMAX1	$\left(\frac{\partial y}{\partial \zeta}\right)$	Partial derivatives of final $y = (r^T, v^T, u^T, \dot{u}^T)^T$ with respect to JMAX1 independent variables
JMAX1	JMAX	Number of independent variables
JLAST	-	JMAX1 + 1
MODE	-	Flag to denote fixed terminal time mission
TIMES(6)	t_f	Terminal time
TT	T	Target epoch (time at which $x_T(T)$ is valid)
XTF(I) for I=1 to 6	$x_T(T)$	Target state at time T
WT(I) for I=1 to 6	w	Diagonal components of weighting matrix ranging from 0.0 to 1.0. (W(I)=1.0 if the Ith end condition is a hard constraint. W(I)=0.0 if the Ith end condition is unconstrained.)

Output Parameter	Symbol	Definition
D(I) for I=1 to 3	h	Orbital angular velocity
for I=4 to 6	e	Eccentricity vector
DELTC(I) for I=1 to 6	Δc	Miss in end conditions
XTF(I)	$x_T(t_F)$	Target state at t_F
DC(I) for I=1 to 6	DC	Weighted combination of transversality conditions and misses in end conditions
E(I,J) for I=1,6 J=1,JMAX1	$\left(\frac{\partial DC}{\partial \zeta}\right)$	Partial derivatives of S with respect to independent variables

C. Method of Computation

Components of the orbital constants h and v are calculated using the expressions

$$h = r \times v$$
$$e = -\left\{\frac{r}{|r|} + \frac{(r \times v) \times v}{\mu}\right\} \quad (1)$$

The subroutine COAST is called to propagate $x_T(T)$ from T to t_f. If a fixed terminal time mission is being flown (indicated by MODE=3), the parameters JMAX1 and JLAST are each decremented by 1. This has the effect of eliminating the dependent variable corresponding to the change in the transversality variable across the last burn arc. It also has the effect of eliminating terminal time as an independent variable and of eliminating the appropriate row and column of the E matrix.

The end condition miss vector Δc is composed of scaled components of Δh, Δe and Δr lying along the R and $K = \frac{H \times R}{|H|}$ vectors and a scaled miss in orbital energy E.

$$\Delta c = \begin{pmatrix} \Delta h^T K/|H| \\ \Delta E\left(\frac{R^2}{\mu}\right) \\ \hline \Delta e^T K \\ \Delta h^T R/|H| \\ \Delta e^T R \\ \Delta r^T K \end{pmatrix} \quad (2)$$

Here, $\Delta h = h_{target} - h$

$\Delta e = e_{target} - e$

and Δc is evaluated at R = r and H = h. This constraint formulation has excellent convergence properties for well posed orbit injection and rendezvous missions of all geometries. All components of Δc are scaled to have the same units as r.

In order to avoid stability problems during the last leg of a mission, the problem is formulated so that a weighted combination of fuel use and miss in end conditions is minimized. The cost functional

$$J = \int_{t_0}^{t_f} |\dot{m}|dt + 1/2\ \Delta c^T W \Delta c \tag{3}$$

is minimized. Here W is a 6×6 diagonal weighting matrix and $|\dot{m}|$ is the rate of fuel consumption during burns. Minimizing this cost functional is equivalent to satisfying the costate equations

$$p_f = \left(\frac{\partial \Delta c}{\partial x}\right)^T W\Delta c \tag{4}$$

where $p_f{}^T = (\dot{u}^T, -u^T)$ or equivalently the equations

$$(I-w)B^T p_f = w\Delta c\Big|_{X=x} \tag{5}$$

where B is a nonsingular matrix such that

$$B(x)^T \left(\frac{\partial \Delta c}{\partial x}\right)^T\Bigg|_{X=x} = I \tag{6}$$

and w is a diagonal weighting matrix with ith diagonal component w_i related to ith diagonal component W_i of W by

$$w_i = \frac{W_i}{1+W_i} \tag{7}$$

Whenever an end condition such as phasing is unconstrained, the corresponding diagonal component of w is zero. For hard constraints, w=I, the vector $B^T p_f$ is composed of six scaled transversality conditions. The sixth component of $B^T p_f$ is $|r|T_v/|h|$ where T_v is the phasing transversality condition calculated in BUZZ. The components of $(1-w)B^T p_f$ given in terms of multiplying coefficients C_{ij} defined in the code are

$$(I-w)B^T p_f = \begin{pmatrix} C_{11}(h^T u) \\ -C_{21}(r^T \dot{u}) - C_{22}(v^T u) \\ -C_{31}(r^T \dot{u}) + C_{32}(r^T u) - C_{33}(v^T u) \\ -C_{41}(h^T \dot{u}) + C_{42}(h^T u) \\ C_{51}(r^T \dot{u}) + C_{52}(r^T u) + C_{53}(v^T u) \\ C_{61}{}^T v \end{pmatrix} \tag{8}$$

The DC vector calculated in BVAL5 corresponds to the miss in satisfying Eq. (5)

$$DC = w\Delta c - (I-w)B^T p_f \tag{9}$$

Partial derivatives of DC with respect to the independent variables ζ are calculated via the chain rule.

$$\left(\frac{\partial DC}{\partial \zeta}\right) = \left(\frac{\partial DC}{\partial x}\right) \left(\frac{\partial x}{\partial \zeta}\right) + \left(\frac{\partial DC}{\partial q}\right) \left(\frac{\partial q}{\partial \zeta}\right) \tag{10}$$

The G matrix in BVAL5 corresponds to (DC/ x) neglecting derivatives of scaling factors. From Eq. (8), it can be seen that the second term in Eq. (10) is efficiently evaluated by calculating terms such as $h^T\left(\frac{\partial u}{\partial \zeta}\right)$, $r^T\left(\frac{\partial \dot{u}}{\partial \zeta}\right)$ and multiplying by the appropriate C_{ij} coefficients.

```
C
C     THIS FILE CONTAINS SUBROUTINES BVAL5, SOLVE, AND BVAL4
C      TO BE USED AS A PART OF GUIDE 71/5 AND GUIDE 71/6.
C
C
C <><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
C
C
C          SUBROUTINE BVAL5 CALCULATES D (ANGULAR MOMENTUM AND
C       ECCENTRICTY-VECTORS) FROM INPUT STATE XF.  IF NBVAL DOES NOT
C       EQUAL -1, THEN THE DC VECTOR ( WEIGHTED COMBINATIONS OF
C       TRANSVERSALITY AND MISS IN END CONDITIONS) AND THE E MATRIX
C       (PARTIAL DERIVATIVES OF DC WITH RESPECT TO THE JMAX1
C       INDEPENDENT VARIABLES) ARE ALSO CALCULATED.
C
C
      SUBROUTINE BVAL5(XF,QF,PTV,TV,NBVAL)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION XF(6),DUM1(6,6),R(3),V(3),DUM2(6)
      COMMON /BVLOUT/XTF(6),DELTC(6)
      COMMON /GIDIN/ XT(6),TT,X0(6),T0,AM0,VEH(10,7),Q0(6),
     1   TIMES(6),C(6)
      COMMON /CPHYS/ UK,REARTH,RHO0,AK,OMEGA,DDLATL
      COMMON /GIDOUT/ DQ0(6),DTIMES(6),E(12,12),DC(12),Y(12),Z(12,12),
     1   D(6),DUMM(4),SM
      COMMON /CINDEX/ NARC,IARC,JMAX,JM,JMAX1,JLAST,NO,NOP,NRKGOS
      COMMON /CMODE/ MODE,IFREZ,ISTOP
      COMMON /CWT/ WT(6)
      DIMENSION G(6,6),U(3),UD(3),RXU(3),RXUD(3),QF(6),PTV(1 ),XK(3)
      DIMENSION VXU(3),VXUD(3),DRU(12),DRUD(12),DVU(12),DHU( 12)
      IF (MODE.NE.3) GO TO 1
      JMAX1=JMAX1-1
      JLAST=JLAST-1
      E(JMAX1,13)=0.
    1 NO=-1
C          SUBROUTINE COAST IS CALLED TO PROPAGATE TARGET STATE TO
C        FINAL TIME SO THAT THE PHASING MISS COMPONENT IN DC(6)
C        CAN BE CALCULATED.
      CALL COAST(XT,DUM2,TIMES(6)-TT,XTF,DUM2,DUM1,DUM1)
      DO 2 I=1,3
      R(I)=XF(I)
      U(I)=QF(I)
      UD(I)=QF(I+3)
    2 V(I)=XF(I+3)
      R2=1.0/(R(1)*R(1)+R(2)*R(2)+R(3)*R(3))
      RM=DSQRT(R2)
      V2=V(1)*V(1)+V(2)*V(2)+V(3)*V(3)
      RTV=R(1)*V(1)+R(2)*V(2)+R(3)*V(3)
      RTVU=RTV/UK
      RTV2=RTV*RTV
      V2U=V2/UK
C          CALCULATE ANGULAR MOMENTUM VECTOR H.
      D(1)=R(2)*V(3)-R(3)*V(2)
      D(2)=R(3)*V(1)-R(1)*V(3)
      D(3)=R(1)*V(2)-R(2)*V(1)
```

```
C     CALCULATE ECCENTRICITY VECTOR E.                                 BVA00560
      DO 3 I=1,3                                                       BVA00570
    3 D(I+3)=-(RM-V2U)*R(I)-RTVU*V(I)                                  BVA00580
      IF (NBVAL.EQ.-1) RETURN                                          BVA00590
      H2=V2/R2-RTV2                                                    BVA00600
      HM=DSQRT(H2)                                                     BVA00610
      CF=0.5*V2-UK*RM                                                  BVA00620
      VUR=V2-RM*UK                                                     BVA00630
      DO 4 I=1,3                                                       BVA00640
    4 XK(I)=(V(I)/R2-RTV*R(I))/HM                                      BVA00650
      HU=HM*UK                                                         BVA00660
      UR2=UK*R2                                                        BVA00670
      URMG=UK/RM-RTV2                                                  BVA00680
      H2MG=H2-RTV2                                                     BVA00690
      CF=0.5*V2-UK*RM                                                  BVA00700
      HC=HM*CF                                                         BVA00710
C     CALCULATE REQUIRED DOT AND CROSS PRODUCTS.                       BVA00720
      RTU=R(1)*U(1)+R(2)*U(2)+R(3)*U(3)                                BVA00730
      RTUD=R(1)*UD(1)+R(2)*UD(2)+R(3)*UD(3)                            BVA00740
      VTU=V(1)*U(1)+V(2)*U(2)+V(3)*U(3)                                BVA00750
      HTU=D(1)*U(1)+D(2)*U(2)+D(3)*U(3)                                BVA00760
      HTUD=D(1)*UD(1)+D(2)*UD(2)+D(3)*UD(3)                            BVA00770
      RXU(1)=R(2)*U(3)-R(3)*U(2)                                       BVA00780
      RXU(2)=R(3)*U(1)-R(1)*U(3)                                       BVA00790
      RXU(3)=R(1)*U(2)-R(2)*U(1)                                       BVA00800
      RXUD(1)=R(2)*UD(3)-R(3)*UD(2)                                    BVA00810
      RXUD(2)=R(3)*UD(1)-R(1)*UD(3)                                    BVA00820
      RXUD(3)=R(1)*UD(2)-R(2)*UD(1)                                    BVA00830
      VXU(1)=V(2)*U(3)-V(3)*U(2)                                       BVA00840
      VXU(2)=V(3)*U(1)-V(1)*U(3)                                       BVA00850
      VXU(3)=V(1)*U(2)-V(2)*U(1)                                       BVA00860
      VXUD(1)=V(2)*UD(3)-V(3)*UD(2)                                    BVA00870
      VXUD(2)=V(3)*UD(1)-V(1)*UD(3)                                    BVA00880
      VXUD(3)=V(1)*UD(2)-V(2)*UD(1)                                    BVA00890
C        CALCULATE REQUIRED COEFFICIENTS.  C COEFFICIENTS MULTIPLY     BVA00900
C      DOT PRODUCTS OF STATE AND COSTATE IN TRANSVERSALITY CONDITIONS. BVA00910
C      B COEFFICIENTS ARE SCALAR MULTIPLIERS IN PARTIALS OF DC WITH    BVA00920
C      RESPECT TO R AND V.                                             BVA00930
      B11=WT(1)*RTV/H2                                                 BVA00940
      B12=WT(1)/(H2*R2)                                                BVA00950
      C11=(1.0-WT(1))*R2                                               BVA00960
      B21=WT(2)*RM                                                     BVA00970
      B22=WT(2)/UR2                                                    BVA00980
      C21=(1.0-WT(2))*UR2/CF                                           BVA00990
      C22=0.5*C21                                                      BVA01000
      U3=(1.0-WT(3))*UR2/HC                                            BVA01010
      BFAC=U3*(RTUD+0.5*VTU)                                           BVA01020
      B31=WT(3)*URMG/HU-BFAC                                           BVA01030
      B32=WT(3)*RTV*VUR/HU+U3*RTU*UR2*RM                               BVA01040
      B33=WT(3)*RTV/(HU*R2)-U3*RTU                                     BVA01050
      B34=WT(3)*H2MG/HU-BFAC                                           BVA01060
      C31=U3*RTV                                                       BVA01070
      C32=(1.0-WT(3))*UR2/HM                                           BVA01080
      C33=0.5*C31                                                      BVA01090
      B41=WT(4)/HM                                                     BVA01100
```

```
      B42=2.0*(1.0-WT(4))*R2/HM
      B43=0.5*B42
      C41=(1.0-WT(4))/HM
      C42=B43*RTV
      U5=(1.0-WT(5))*UR2/(H2*CF)
      BFAC=U5*RTV*(2.0*RTUD+VTU)
      B51=WT(5)*V2U-U5*(V2*RTUD-UK*RTV*RTU*R2*RM-0.5*UK*VTU*HM)
      B52=WT(5)*RTVU+U5*CF*RTU-BFAC
      B53=2.0*WT(5)/UR2-U5*(CF*RTU-RTUD/R2)
      B54=-2.0*WT(5)*RTVU+BFAC
      C51=0.5*U5*H2MG
      C52=U5*RTV*CF
      C53=0.5*U5*URMG
      B61=WT(6)*RM
      C61=(1.0-WT(6))/(RM*HM)
      C62=(1.0-WT(6))*UR2/HM
C        CALCULATE PARTIALS OF DC WITH RESPECT TO R AND V.
      DO 5 I=1,3
      G(1,I)=B11*D(I)-C11*VXU(I)
      G(1,I+3)=-B12*D(I)+C11*RXU(I)
      G(2,I)=B21*R(I)+C21*UD(I)+(C21*RTUD+C22*VTU)*UR2*RM*R(I)/CF
      G(2,I+3)=B22*V(I)+C22*U(I)+(C21*RTUD+C22*VTU)*V(I)/CF
      G(3,I)=-B31*V(I)-B32*R(I)-C32*U(I)+C31*UD(I)
      G(3,I+3)=-B33*V(I)-B34*R(I)+C33*U(I)
      G(4,I)=-B41*D(I)+B42*R(I)-B43*V(I)+C41*VXUD(I)+C42*VXU(I)
      G(4,I+3)=-B43*R(I)-C41*RXUD(I)-C42*RXU(I)
      G(5,I)=B51*R(I)-B52*V(I)-C51*UD(I)-C52*U(I)
      G(5,I+3)=B53*V(I)+B54*R(I)+C53*U(I)
      G(6,I)=B61*XK(I)-C61*PTV(I)
    5 G(6,I+3)=-C61*PTV(I+3)
      DO 6 J=1,JMAX1
      DRU(J)=R(1)*Z(7,J)+R(2)*Z(8,J)+R(3)*Z(9,J)
      DRUD(J)=R(1)*Z(10,J)+R(2)*Z(11,J)+R(3)*Z(12,J)
      DVU(J)=V(1)*Z(7,J)+V(2)*Z(8,J)+V(3)*Z(9,J)
      DHU(J)=D(1)*Z(7,J)+D(2)*Z(8,J)+D(3)*Z(9,J)
C          CALCULATE PARTIALS OF DC WITH RESPECT TO COSTATE TIMES
C       PARTIAL OF COSTATE WITH RESPECT TO INDEPENDENT VARIABLES.
C       (FIRST STEP OF CHAIN RULE)
      E(1,J)=-C11*DHU(J)
      E(2,J)=C21*DRUD(J)+C22*DVU(J)
      E(3,J)=C31*DRUD(J)-C32*DRU(J)+C33*DVU(J)
      E(4,J)=C41*(D(1)*Z(10,J)+D(2)*Z(11,J)+D(3)*Z(12,J))-C42*DHU(J)
      E(5,J)=-C51*DRUD(J)-C52*DRU(J)-C53*DVU(J)
      E(6,J)=-C61*(V(1)*Z(10,J)+V(2)*Z(11,J)+V(3)*Z(12,J))-C62*DRU(J)
C          ADD IN PARTIAL OF DC WITH RESPECT TO STATE TIMES PARTIAL
C       OF STATE WITH RESPECT TO INDEPENDENT VARIABLES.
      DO 6 I=1,6
      DO 6 K=1,6
    6 E(I,J)=E(I,J)+G(I,K)*Z(K,J)
C         CALCULATE MISS IN SOFT CONSTRAINTS.
C            1.  DELTA H ALONG H CROSS R
C            2.  DELTA ENERGY
C            3.  DELTA E ALONG H CROSS R
C            4.  DELTA H ALONG R
C            5.  DELTA E ALONG R
```

```
C              6.  DELTA R ALONG H CROSS R                                      BVA01660
C       WHERE DELTA REPRESENTS DESIRED MINUS ACTUAL AND CONSTRAINTS             BVA01670
C       ARE SCALED TO HAVE UNITS OF LENGTH.                                     BVA01680
      DELTC(1)=(C(1)*XK(1)+C(2)*XK(2)+C(3)*XK(3))/HM                            BVA01690
      DELTC(2)=-(CF-.5*UK*UK*(C(4)*C(4)                                         BVA01700
     1             +C(5)*C(5)+C(6)*C(6)-1.0)/(C(1)*C(1)                         BVA01710
     1             +C(2)*C(2)+C(3)*C(3)))/UR2                                   BVA01720
      DELTC(3)=(C(4)-D(4))*XK(1)+(C(5)-D(5))*XK(2)+(C(6)-D(6))*XK(3)            BVA01730
      DELTC(4)=(C(1)*R(1)+C(2)*R(2)+C(3)*R(3))/HM                               BVA01740
      DELTC(5)=(C(4)-D(4))*R(1)+(C(5)-D(5))*R(2)+(C(6)-D(6))*R(3)               BVA01750
      DELTC(6)=(XTF(1)*XK(1)+XTF(2)*XK(2)+XTF(3)*XK(3))*RM                      BVA01760
C          CALCULATE WEIGHTED COMBINATIONS OF TRANSVERSALITY                    BVA01770
C         CONDITIONS AND MISS IN END CONDITIONS.                                BVA01780
      DC(1)=C11*HTU+WT(1)*DELTC(1)                                              BVA01790
      DC(2)=-C21*RTUD-C22*VTU+WT(2)*DELTC(2)                                    BVA01800
      DC(3)=-C31*RTUD+C32*RTU-C33*VTU+WT(3)*DELTC(3)                            BVA01810
      DC(4)=-C41*HTUD+C42*HTU+WT(4)*DELTC(4)                                    BVA01820
      DC(5)=C51*RTUD+C52*RTU+C53*VTU+WT(5)*DELTC(5)                             BVA01830
      DC(6)=C61*TV+WT(6)*DELTC(6)                                               BVA01840
      RETURN                                                                    BVA01850
      END                                                                       BVA01860
C                                                                               BVA01870
C <><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>  BVA01880
C                                                                               BVA01890
C                                                                               BVA01900
      SUBROUTINE SOLVE(A,L6,L7)                                                 BVA01910
      REAL*8 A(12,25),Q                                                         BVA01920
      DO 6 N=1,L6                                                               BVA01930
      M=N+1                                                                     BVA01940
      IBIG=N                                                                    BVA01950
      DO 1 I=N,L6                                                               BVA01960
    1 IF( DABS( A(I,N) ).GT. DABS(A(IBIG,N)))  IBIG=I                           BVA01970
      DO 2 J=M,L7                                                               BVA01980
      Q=A(IBIG,J)/A(IBIG,N)                                                     BVA01990
      A(IBIG,J)=A(N,J)                                                          BVA02000
    2 A(N,J)=Q                                                                  BVA02010
      A(IBIG,N)=A(N,N)                                                          BVA02020
      DO 5 I=1,L6                                                               BVA02030
      IF(I.EQ.N) GO TO 5                                                        BVA02040
      DO 4 K=M,L7                                                               BVA02050
    4 A(I,K)=A(I,K)-A(I,N)*A(N,K)                                               BVA02060
    5 CONTINUE                                                                  BVA02070
    6 CONTINUE                                                                  BVA02080
      RETURN                                                                    BVA02090
      END                                                                       BVA02100
C                                                                               BVA02110
C <><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>    BVA02120
C                                                                               BVA02130
      SUBROUTINE BVAL4(XF,JF,PTV,TV,NBVAL)                                      BVA02140
      IMPLICIT REAL*8(A-H,O-Z)                                                  BVA02150
C  THIS IS A FOUR-CONSTRAINT VERSION OF BVAL6.  THE MISSION                     BVA02160
C  IS TO ACHIEVE AN ORBIT WITH GIVEN VALUES OF SEMIMAJOR AXIS,                  BVA02170
C  ECCENTRICITY, INCLINATION, AND ARGUMENT OF PERIGEE.  THE ORBITAL             BVA02180
C  CONSTANTS WHICH ARE TRANSMITTED IN C IN THE COMMON BLOCK GIDIN               BVA02190
C  ARE MAGNITUDE AND THIRD COMPONENT OF ORBITAL ANGULAR VELOCITY H,             BVA02200
C  THE THIRD COMPONENT OF A VECTOR E POINTING TOWARD PERICENTER WITH             BVA02210
```

```
C     MAGNITUDE OF ECCENTRICITY, AND THE THIRD COMPONENT OF H X E.
      COMMON /CPHYS/ UK,REARTH,RHO0,RHOB,OMEGA,OBLATE
      COMMON /CINDEX/ NARC,IARC,JMAX,JM,JMAX1,JLAST,NO,NOP,NOKGOS
      COMMON /GIDIN/ XT(6),TT,X0(6),T0,AM0,VEH(10,7),Q0(6),TIMES(6),C(6)
      COMMON /GIDOUT/ DQC(6),DTIMES(6),E(12,13),ZZ(12),Z(12,12),D(6),
     1 DUMM(4),SM
      DIMENSION XF(6),QF(6),PTV(12),G(6)
C***< 1 > CALCULATE D(I) FOR I = 1 TO 4 ********************* ***********
      R2=1.0/(XF(1)*XF(1)+XF(2)*XF(2)+XF(3)*XF(3))
      RM=(R2)**0.5
      RTV=XF(1)*XF(4)+XF(2)*XF(5)+XF(3)*XF(6)
      V2=XF(4)*XF(4)+XF(5)*XF(5)+XF(6)*XF(6)
      D(1)=(V2/R2-RTV*RTV)**0.5
      D(2)=XF(1)*XF(5)-XF(2)*XF(4)
      H2=D(1)*D(1)
      RTVR=RTV*RM
      FV=H2/UK-1.0/RM
      D(4)=RTVR*XF(3)+FV*XF(6)
      D(4)=RTV*XF(3)*RM-XF(3)/RM+H2*XF(6)/UK
      V2UR=V2/UK-RM
      RTVU=RTV/UK
      D(3)=XF(3)*V2UR-XF(6)*RTVU
      IF (NBVAL.EQ.-1) RETURN
C***< 2 > CALCULATE PARTIAL DERIVATIVES E(I,J) ************* ***********
C.....< 2.1 > CALCULATE (DC4/DR), (DC4/DV) ...............................
      R3R=XF(3)*RM*R2
      V3U=XF(6)/UK
      V2H=V2/D(1)
      RTVH=RTV/D(1)
      R2H=1.0/(R2*D(1))
      TR3=2.0*XF(3)/UK
      CS1=XF(6)*(V2UR+V2/UK)-RTV*R3R
      CS2=2.0*V3U/R2
      F=XF(3)*RM-2.0*RTVU*XF(6)
      DO 4 I=1,3
      G(I)=F*XF(I+3)+CS1*XF(I)
    4 G(I+3)=CS2*XF(I+3)+F*XF(I)
      G(3)=G(3)+RTVR
      G(6)=G(6)+FV
      DO 1 J=1,JMAX1
C.....< 2.2 > CALCULATE R'S AND V'S.......................................
      RZ1=XF(1)*Z(1,J)+XF(2)*Z(2,J)+XF(3)*Z(3,J)
      RZ4=XF(1)*Z(4,J)+XF(2)*Z(5,J)+XF(3)*Z(6,J)
      VZ1=XF(4)*Z(1,J)+XF(5)*Z(2,J)+XF(6)*Z(3,J)
      VZ4=XF(4)*Z(4,J)+XF(5)*Z(5,J)+XF(6)*Z(6,J)
C.....< 2.3 > FINISH CALCULATION OF E(I,J)................................
      E(2,J)=XF(5)*Z(1,J)-XF(4)*Z(2,J)-XF(2)*Z(4,J)+XF(1)*Z(5,J)
      E(1,J)=V2H*RZ1-RTVH*(VZ1+RZ4)+R2H*VZ4
      E(3,J)=V2UR*Z(3,J)+R3R*RZ1+TR3*VZ4-V3U*(VZ1+RZ4)-RTVU*Z(6,J)
      E(4,J)=G(1)*Z(1,J)+G(2)*Z(2,J)+G(3)*Z(3,J)+G(4)*Z(4,J)+
     1 G(5)*Z(5,J)+G(6)*Z(6,J)
      E(5,J)=+QF(5)*Z(1,J)-QF(4)*Z(2,J)-QF(2)*Z(4,J)+QF(1)*Z(5,J)
     1 +XF(5)*Z(7,J)-XF(4)*Z(8,J)-XF(2)*Z(10,J)+XF(1)*Z(11,J)
      SUM=0.0
      DO 2 K=1,12
```

FILE: CASBVJ FORTRAN P1 CAMBRIDGE MONITOR SYSTEM

```
    2 SUM=SUM+PTV(K)*Z(K,J)                                           BVA02760
    1 E(6,J)=SUM                                                      BVA02770
C***< 3 > CALCULATE MISS IN END CONDITIONS DEL C *********************BVA02780
      DO 3 I=1,4                                                      BVA02790
    3 E(I,13)=C(I)-D(I)                                               BVA02800
      E(6,13)=-TV                                                     BVA02810
      E(5,13)=-(XF(5)*QF(1)-XF(4)*QF(2)-XF(2)*QF(4)+XF(1)*QF(5))      BVA02820
      RETURN                                                          BVA02830
      END                                                             BVA02840
```